Deep Asymmetric Metric Learning via Rich Relationship Mining

Xinyi Xu[1], Yanhua Yang[1], Cheng Deng[1]*, Feng Zheng[2]
[1]School of Electronic Engineering, Xidian University, Xian 710071, China
[2]Department of Computer Science and Engineering, Southern University of Science and Technology
xyxu.xd@gmail.com, yanhyang@xidian.edu.cn, chdeng.xd@gmail.com, zhengf@sustc.edu.cn

Abstract

Learning effective distance metric between data has gained increasing popularity, for its promising performance on various tasks, such as face verification, zero-shot learning, and image retrieval. A major line of researches employs hard data mining, which makes efforts on searching a subset of significant data. However, hard data mining based approaches only rely on a small percentage of data, which is apt to overfitting. This motivates us to propose a novel framework, named deep asymmetric metric learning via rich relationship mining (DAMLRRM), to mine rich relationship under satisfying sampling size. DAMLRRM constructs two asymmetric data streams that are differently structured and of unequal length. The asymmetric structure enables the two data streams to interlace each other, which allows for the informative comparison between new data pairs over iterations. To improve the generalization ability, we further relax the constraint on the intra-class relationship. Rather than greedily connecting all possible positive pairs, DAMLRRM builds a minimum-cost spanning tree within each category to ensure the formation of a connected region. As such there exists at least one direct or indirect path between arbitrary positive pairs to bridge intra-class relevance. Extensive experimental results on three benchmark datasets including CUB-200-2011, Cars196, and Stanford Online Products show that DAMLRRM effectively boosts the performance of existing deep metric learning approaches.

1. Introduction

Metric learning aims at finding appropriate similarity measurements of data, whose major thinking is to keep the distance between similar instances close and dissimilar instances far away in an embedding space. This topic is of great practical importance due to its wide applications, including face recognition [12, 52, 45], clustering [9, 44, 53], and retrieval [57, 49, 51, 50, 22, 10]. Conventional Mahalanobis metric learning approaches learn a linear transformation of the data and measure the similarity based on Euclidean distance, which fail to capture the high-order correlation[15, 42, 47]. Riding on the development of deep neural network [21, 33, 37], deep metric learning (DML) has gained a lot of attention. Guided by a metric loss, DML projects data into an embedding space with rich semantic information through convolutional neural network. It shows potential capability even in challenging tasks, such as fine-grained classification [8, 41, 55, 25], large-category classification [2, 31, 46], and zero-shot learning [28, 56, 6, 26].

According to the types of loss, DML can be roughly divided into contrastive and triplet approaches. However, enumerating all possible pairs or triplets will arise nearly exponential sampling size, which is impractical even for a moderate number of instances. One common solution is to sample a subset of instances as a training pool. The fact is that, when the sampled training pool merely covers easy instances that contribute little to the optimization, only a weak embedding model can be obtained. Therefore, hard data mining aiming to find out confusing instances becomes an important topic, and a large number of methods are proposed [35, 34, 16, 11, 43, 54, 14]. Those methods tackle this topic to a certain extent yet are still deficient in the following three aspects. First, a complicated data preprocessing is involved to select hard data, whereas the hard level is changing with the evolution of the model [34]. Second, only a small subset of relationship is exploited. Third, the hard level is difficult to control. When the selected instances are not hard enough, the learned model is not discriminative. Conversely, when the instances are selected too hard, the overfitting problem often occurs [43].

In this work, we propose a novel framework, named deep asymmetric metric learning via rich relationship mining (DAMLRRM). DAMLRRM firstly builds two asymmetric data streams, which interlace to each other so that continues new pairs are compared during iterations. Compared with conventional one stream metric learning approaches, DAMLRRM can mine considerably richer relationship un-

*Corresponding author.

der lower sampling size. Furthermore, DAMLRRM relaxes the constraint on positive pairs to extend the generalization capability. Specifically, we build positive pairs training pool by constructing a minimum connected tree for each category instead of considering all positive pairs within a mini-batch. As a result, there will exist a direct or indirect path between any positive pair, which ensures the relevance being bridged to each other. The inspiration comes from ranking on manifold [58] that spreads the relevance to their nearby neighbors one by one. The connected graph loss can help maintain the inherent distribution of the data and achieve a good generalization ability. In experiments, we empirically show the state-of-the-art results on CUB200-2011 [39], Cars196 [1], and Stanford online products [35] datasets for clustering and retrieval tasks. In a nutshell, this paper makes the following contributions:

i) We departure from the traditional hard data mining based technique and propose a novel asymmetric two streams based deep learning framework for metric learning, which also differs from conventional methods only involving one stream.

ii) We devise a relaxation technique for positive pairs constraint to improve the model generalization ability, which is verified in our empirical study.

iii) Our proposed model achieves better accuracy when using fewer than ten percents of sampling size compared with the peer methods including the lifted method [35] and N-pair [34].

2. Related Work

Siamese network [5] is the seminal work of the contrastive DML. It firstly employs twins networks to nonlinearly map two signature instances into feature space. And subsequently, a contrastive loss is employed to optimize the mapping procedure. The contrastive loss minimizes the distance between positive pairs and enlarges the distance between negative pairs if they are closer than a predetermined margin. Based on the siamese network, a collection of approaches are proposed to settle dimensionality reduction and face verification tasks [13, 7, 36, 38].

Although making great progress, contrastive metric learning approaches suffer from one drawback, that focus on absolute distance whereas relative distance matters more for most tasks [30, 31, 35]. Triplet loss, an evolution formulation of contrastive loss, has been proposed to tackle this issue. It trains a model on a triplets training pool, where each triplet consists of an anchor, a positive and a negative instance. The anchor and the positive instances share the same label, while the anchor and the negative instances have different labels. The training process encourages the network to find an embedding where the distance between positive pairs is smaller than the distance between negative pairs with some margins.

Nevertheless, contrastive and triplet losses tend to be difficult to optimize in practice, mainly influenced by the way of selecting the training pool. Confusing instances, doing a crucial contribution to optimization, should be paid huge attention to. FaceNet [31] targets on the online hard data generation, which uses large mini-batches in the order of a few thousand instances and only computes the argmin and argmax within a mini-batch. However, the batch size is 1800, which is a big memory obstacle when implementation. To take full advantage of relative relationship, Song *et al.* [35] allow mining the negatives from both the left and right data pair instead of negative being defined only according to anchor points. Chen *et al.* [16] introduce a position-dependent deep metric unit, which can be used to select hard instances to guide the deep embedding learning in an online and robust manner. Sohn *et al.* [34] indicate that a minority of negative instance based loss function suffers from poor local optima. As a result, they propose an $(N+1)$-tuplet loss that optimizes to identify a positive instance from $N-1$ negative instances, which gains some performance improvement. More recently, Duan *et al.* [11] propose a deep adversarial metric learning framework to generate synthetic hard negatives from the observed negative instances.

The fundamental philosophy behind hard data mining is that for a pair of positive instances, select a significant negative sample through offline or online and penalize on the relative distance if they violate the constraint. However, both offline-based and online-based hard data mining strategies exist defects. Offline-based methods select the hard instances before training which will not be updated with the updating models. It is unreasonable as a hard relationship is dynamically decided by different models. Online-based methods decide the hard negative sample within a mini-batch along with training, which makes the comparison within a very small subset of instances. The hard quality is not guaranteed. One common drawback of these two forms is that the learned metric is insufficient because of the low utilization of pairs or triplets. Therefore, we make efforts on exploiting more pairs while controlling the sampling size in this paper.

Graph is a mathematical structure used to model pairwise relations between objects [4, 3]. A graph in the context is made up of vertices and points which are connected by edges. Graph knowledge is used to express the correlation network in many applications, such as image retrieval [32], Linguistics processing[17] and saliency detection [48]. More recently, Iscen *et al.* [18] utilize an undirected graph to mine an efficient training pool without label, which verifies the priority of graph in building correlation. In this paper, we take advantage of the graph to relax the constraint between positive pairs, which is quite helpful to boost the generalization ability.

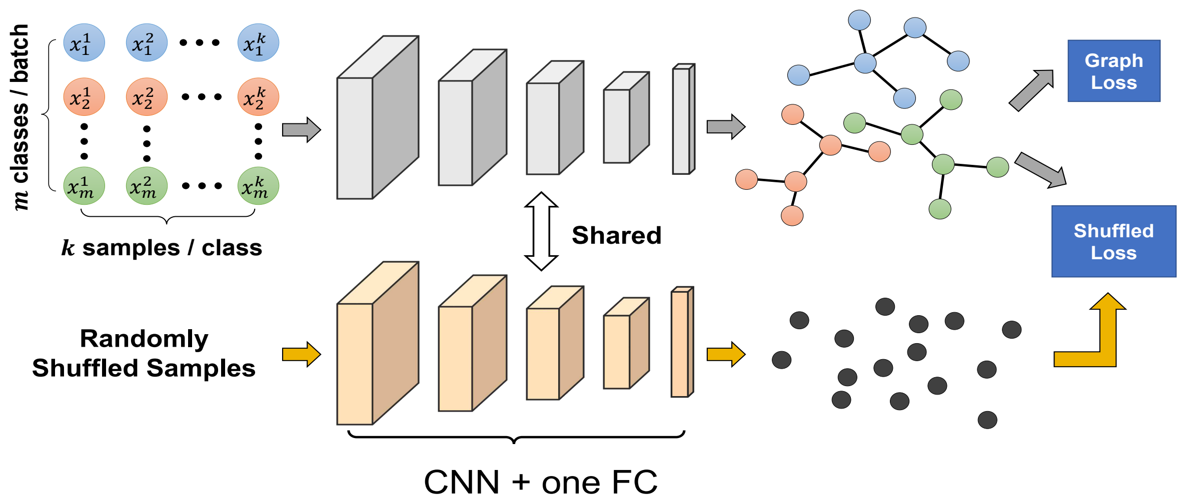


Figure 1. The asymmetric deep metric learning framework of our proposed method. Two different structured batches are employed for two stream network training, where the upper one is neatly arranged and the lower one is shuffled. Mapped by two shared networks, the feature embedding of instances are obtained and the learning process is supervised by two loss functions.

3. Proposed Approach

Figure 1 illustrates the framework of our proposed method DAMLRRM. Two weight-shared networks are employed to map two asymmetric data batches, where the upper stream accepts neatly arranged data (neat stream) and the lower stream takes shuffled data as input (shuffled stream). Our model builds a minimum-cost spanning tree for each class in the neat stream which establishes a stable intra-class manifold. Furthermore, the strong discrimination capability is achieved by adopting a shuffled stream to provide various negative instances for the neat stream. We detail our proposed model in the following subsections.

3.1. Preliminaries

Let $\mathbf{X} = \{\mathbf{x}_i | i = 1, 2, \cdots, N^x\}$ and $\mathbf{S} = \{\mathbf{s}_i | i = 1, 2, \cdots, N^s\}$ be the training pools of two streams, where N^x and N^s are the numbers of instances in $\mathbf{X}$ and $\mathbf{S}$ respectively. The target of DML is to learn a nonlinear transformation to semantic embedding space $f : R^{\widetilde{d}} \rightarrow R^d$, which is a differentiable deep network with parameter θ. We measure the similarity of $(\mathbf{x}_i, \mathbf{x}_j)$ in term of the Euclidean distance in the embedding space, which is computed as $\mathbf{D}_{ij} = \|f(\mathbf{x}_i) - f(\mathbf{x}_j)\|^2$. Furthermore, we construct each category as an undirected weighted subgraph $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{D})$, where each node in $\mathbf{V}$ corresponds to a sample, the edges in $\mathbf{E}$ connect positive pairs, and $\mathbf{D}$ stores the edge weights.

3.2. Rich Relationship Mining with Asymmetric Structure

To obtain rich relationship, we propose an asymmetric framework for metric learning. Asymmetry is reflected in structure and quantity, respectively. In structure, two total different structured data batches are built for two streams respectively. The data batch of the upper stream is neat while the other one is randomly shuffled. It can be clearly shown on the left side of Figure 1 and the formulations are

$$\begin{aligned} \mathbf{B}^x &= \{\mathbf{x}_1^1, \cdots, \mathbf{x}_1^k; \mathbf{x}_2^2, \cdots, \mathbf{x}_2^k; \cdots; \mathbf{x}_m^1, \cdots, \mathbf{x}_m^k\} \\ \mathbf{B}^s &= \{\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_{b/2}\} \\ \mathbf{B} &= \{\mathbf{B}^x, \mathbf{B}^s\}. \end{aligned} \tag{1}$$

The neat data batch $\mathbf{B}^x$ is composed of m categories where there are k instances for each category. And the shuffled data batch $\mathbf{B}^s$ contains $b/2$ randomly instances, where $b/2 = m * k$. Hence for each iteration, the training batch $\mathbf{B}$ consists of two parts: one neat data batch $\mathbf{B}^x$ and one shuffled data batch $\mathbf{B}^s$.

In quantity, the training pools' sizes of the two streams are unequal, namely $N^x \neq N^s$. Quantity asymmetric makes it possible that the same instances in one data stream compare with different instances in another data stream at different iteration times. For example, $\mathbf{B}_1^s$, $\mathbf{B}_l^s$ and $\mathbf{B}_n^s$ in Figure 2 include the same instances at different iteration times, while they compares with different instances in stream 1. Specifically, $\mathbf{B}_1$ and $\mathbf{B}_l$ are composed by

$$\begin{aligned} &\mathbf{B}_1 = \{\mathbf{B}_1^x;\ \mathbf{B}_1^s\}, \quad \mathbf{B}_l = \{\mathbf{B}_1^x;\ \mathbf{B}_l^s\} \\ &\mathbf{B}_1^x \neq \mathbf{B}_l^x, \quad \mathbf{B}_1^s = \mathbf{B}_l^s. \end{aligned} \tag{2}$$

By doing so, our model can exploit abundant relationship while not increase the sampling size.

The intuitive motivations behind the asymmetric metric learning come from two aspects: 1) The neat stream mainly focus on establishing the consistent intra-class relationship by a minimum-cost spanning tree, which constrains positive

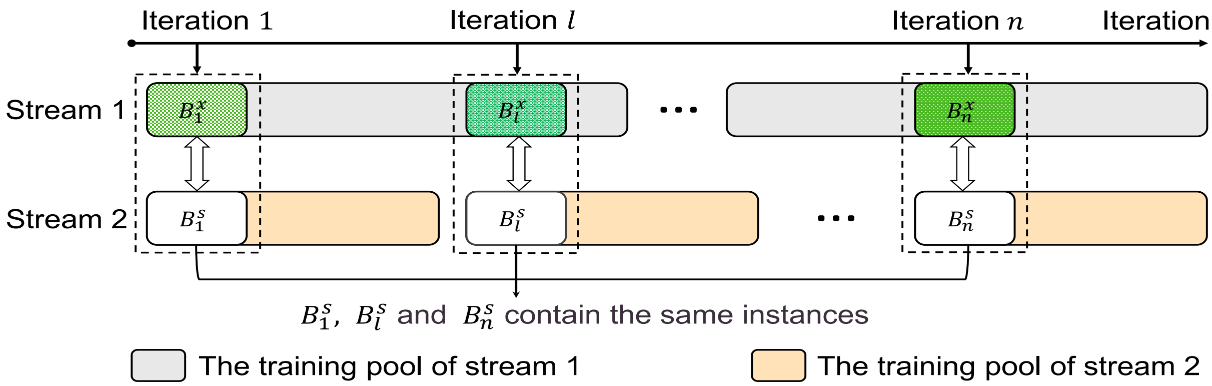


Figure 2. The interlaced batches. The lengths of the two data streams are various, which make the same instances contained by stream2 interacting with different instances in stream 1. Under the same sampling size, such interlaced batches can mine more pair correlation.

pairs to form a unified manifold. However, such an intra-class relationship is not stable enough because it observes limited negative instances; 2) the shuffled stream generates diverse and numerous negative instances for the neat stream, which aims to establish a discriminative inter-class relationship. It is worth note that, the two batches do not cause extra memory or computation cost, because we just split half of the batch size which previous approaches employ and the two networks share all weights.

3.3. Connected Graph based Loss Functions

Previous approaches constrain on all possible positive pairs in a mini-batch, which is too strict and causes an overfitting problem. Inspired by the method of ranking on data manifold [58], that the global consistency is obtained by spreading the relevance of source point to its nearest neighbors one by one, we relax the constraint of positive pairs. Rather than connecting all positive pairs, we build a minimum-cost spanning tree for each category. By doing so, a connected field within one class is obtained, which ensure a direct or indirect path exists between arbitrary positive pairs and not too much pressure is employed on the pairs which are not visually similar. In other words, the instances that far distributed in the original visual space are allowed indirectly associated and their distance being larger than the threshold. The central idea is to retain the intrinsic distribution of data to the utmost extent while ensuring semantic consistency.

We employ a simple minimum-spanning tree algorithm named *prim* [27] to build the connected graph. *Prim* algorithm is a greedy algorithm which finds a minimum spanning tree for a weighted undirected graph. It finds a subset of the edges to form a tree which includes every vertex, where the total weight of all edges in the tree is minimized. The procedure of *prim* algorithm is summarized as follows:

(1) Build a weighted graph $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{D})$, where $\mathbf{D}$ is measured by Euclidean distance. Set $\mathbf{V}_{visted} = \{\emptyset\}$ and $\mathbf{V}_{unvisted} = \mathbf{V}$. Initialize a tree with a single arbitrary vertex $\mathbf{V}_{start} \in \mathbf{V}$. Add $\mathbf{V}_{start}$ into $\mathbf{V}_{visted}$ and remove it from $\mathbf{V}_{unvisted}$.

(2) Grow the tree by one edge: choosing a minimum-weight edge $\mathbf{E}_{minimum} \in \mathbf{E}$ which connect $\mathbf{V}_{visted}$ and $\mathbf{V}_{unvisted}$, then attach it to the tree. Add the minimum-weight-connected vertex into $\mathbf{V}_{visted}$ and remove it from $\mathbf{V}_{unvisted}$.

(3) Repeat step 2 until all vertices are covered in the tree ($\mathbf{V}_{visted} = \mathbf{V}$).

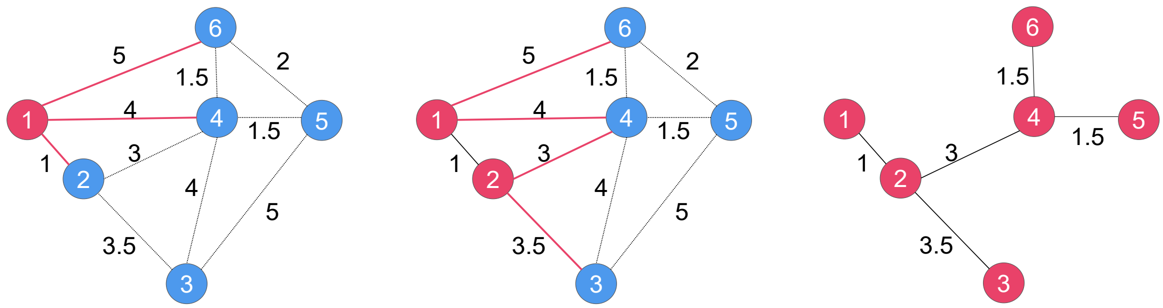
Figure 3 gives a concrete example. Suppose the starting vertex being the point 1 (Figure 3(a)), then the next vertex will reach point 2 (Figure 3(b)) by choosing the minimum weight connected to point 1. Then find out the minimum weight of all edges connected to both point 1 and 2 and hence reached point 4. Repeat this progress until all vertices are included in the tree like Figure 3(c). For the situation in Figure 3,

$$PP = \{(\mathbf{x}_1, \mathbf{x}_2); (\mathbf{x}_2, \mathbf{x}_3); (\mathbf{x}_2, \mathbf{x}_4); (\mathbf{x}_4, \mathbf{x}_5); (\mathbf{x}_4, \mathbf{x}_6)\}, \tag{3}$$

where PP is the connected positive pairs pool. Notably, this minimum-cost spanning tree is quite different from simply choosing the nearest positive pairs which does not ensure a connected field within a category.

The objective function is defined based on the built positive pairs pool. Predefine a boundary α and a margin β, the optimize goal is limiting the distance of positive pairs smaller than $\alpha - \beta$. For the negative instances, we hope they will not break into the tree, so the distance is forced to be bigger than $\alpha + \beta$. The graph loss function is defined as

$$\mathcal{L}^g = \frac{1}{P_g} \left(\sum_{i,j \in PP} [\mathbf{D}_{i,j} - \alpha + \beta]_+^2 + \sum_{i,j \in NP} [-\mathbf{D}_{i,j} + \alpha + \beta]_+^2 \right), \tag{4}$$



(a) An Undirected Weighted Graph (b) An Undirected Weighted Graph (c) Minimum-Cost Spanning Tree

Figure 3. The building procedure of minimum-cost spanning tree. The relevance is bridged from one to another at the smallest weight cost until all instances within this category are connected directly or indirectly.

where P_g is the number of pairs that violate the constraint and NP is the negative pairs pool.

For the instances in the shuffled stream, the positive instances are expected to join into the *prim* tree, we force it to connect to the nearest point in the tree. And the negative instances are constrained to be far away. Hence the shuffled loss $\mathcal{L}^s$ is defined as

$$\mathcal{L}^s = \frac{1}{P_s}\left(\sum_{i,j\in NN}[\mathbf{D}_{i,j}-\alpha+\beta]_+^2 + \sum_{i,j\in NP}[-\mathbf{D}_{i,j}+\alpha+\beta]_+^2\right), \tag{5}$$

where P_s is the number of violated pairs and NN is the top 1 nearest positive pairs pool between two streams.

Combining the two loss functions, the final objective function can be formulated as:

$$\mathcal{L} = \mathcal{L}^g + \mathcal{L}^s, \tag{6}$$

where we do not employ any balance parameter when combining the two loss functions, as the motivation behind them are the same. The principle that we obey to design the objective function is respecting the data distribution most as long as the semantic consistency satisfied. The goal, accomplished by connecting connect the nearest positive pairs of the two stream and build *prim* tree within a neat stream, is to relax the constraint and achieve generalization ability.

4. Experiments

In this section, we evaluate the effectiveness of our proposed DAMLRRM on three public benchmark datasets for both image retrieval and clustering tasks. The Caffe package [20] is used through the experiments. All images are resized to 256-by-256 at first. For data augmentation, the training instances are performed standard random crop and horizontal mirroring, while a single center crop for testing. The embedding size is set to $d = 512$ for all embedding vectors [40, 11]. GoogLeNet [37] pretrained on ImageNet ILSVRC dataset [29] is used for initialization and a randomly initialized fully connected layer is added. The base learning rate is set to $10e - 4$ and 10 times faster for the newly added fully connected layer. We use SGD with $40k$ training iterations and 60 mini-batch size for each stream.

4.1. Benchmark Datasets

We conduct our experiments on CUB-200-2011 [39], Cars196 [1] and Stanford Online Products [35] . For all datasets, we follow the conventional protocol of splitting training and testing [35]:

CUB-200-2011 [39] dataset covers 200 species of birds with $11,788$ instances, where the first 100 species ($5,864$ images) are used for training and the rest of 100 species ($5,924$ images) are used for testing.

Cars196 [1] dataset is composed by $16,185$ cars images of 196 classes. We use the first 98 classes ($8,054$ images) for training and the other 98 classes ($8,131$ images) for testing.

Stanford Online Products [35] dataset contains $22,634$ classes with $120,053$ product images in total, where the first $11,318$ classes ($59,551$ images) are used for training and the remaining $11,316$ classes ($60,502$ images) are used for testing.

When building the tree in the neat stream, we set $k = 5$ for CUB-200-2011 and Cars196 , and $k = 3$ for Stanford Online Products because each product has only about 5.3 images.

4.2. Baselines

To verify the superiority of our proposed method, we compare with eight baseline deep metric learning algorithms, which are 1) DDML [23]; 2) contrastive embedding loss (Contrastive) [13]; 3) Triplet embedding loss (Triplet) [42]; 4) triplet loss with N-pair sampling, (Triplet+N-pair); 5) Lifted [35]; 6) N-pair loss (N-pair) [34]; 7) Angular loss (Angular) [40]; and 8) adversarial metric loss (AML) [11].

As the central issue of this work is the sufficient relationship mining under a small sampling size, we did not employ any hard negative mining strategies to complicate the com-

Table 1. Comparison of clustering and retrieval on CUB_200_2011 [39] dataset

Method	Clustering(%)		Recall@a(%)			
	NMI	F_1	R@1	R@2	R@4	R@8
DDML[23]	47.3	13.1	31.2	41.6	54.7	67.1
Contrastive[13]	47.2	12.5	27.2	36.3	49.8	62.1
Triplet[42]	49.8	15.0	35.9	47.7	59.1	70.0
Triplet+N-pair	54.1	20.0	42.8	54.9	66.2	77.6
Lifted[35]	56.4	22.6	43.6	56.6	68.6	79.6
N-pair[34]	60.2	28.2	51.9	64.3	74.9	83.2
Angular[40]	61.0	**30.2**	**53.6**	65.0	75.3	83.7
AML[11]	**61.3**	29.5	52.7	**65.4**	**75.5**	**84.3**
OURS	**61.7**	**31.2**	**55.1**	**66.5**	**76.8**	**85.3**

Table 2. Comparison of clustering and retrieval on the Cars196 [1] dataset

Method	Clustering(%)		Recall@a(%)			
	NMI	F_1	R@1	R@2	R@4	R@8
DDML[23]	41.7	10.9	32.7	43.9	56.5	68.8
contrastive[13]	42.3	10.5	27.6	38.3	51.0	63.9
Triplet[42]	52.9	17.9	45.1	57.4	69.7	79.2
Triplet+N-pair	54.3	19.6	46.3	59.9	71.4	81.3
Lifted[35]	55.1	25.1	48.3	61.1	71.8	81.1
N-pair[34]	62.7	31.8	68.9	78.9	85.8	90.9
Angular[40]	62.4	31.8	71.3	80.7	87.0	91.8
AML[11]	**63.1**	**31.9**	**72.5**	**82.1**	**88.5**	**92.9**
OURS	**64.2**	**33.5**	**73.5**	**82.6**	**89.1**	**93.5**

Table 3. Comparison of clustering and retrieval on the stanford online products [35] dataset

Method	Clustering(%)		Recall@a(%)		
	NMI	F_1	R@1	R@10	R@100
DDML[23]	83.4	10.7	42.1	57.8	73.7
Contrastive[13]	82.4	10.1	37.5	53.9	71.0
Triplet[42]	86.3	20.2	53.9	72.1	85.7
Triplet+N-pair	86.4	21.0	58.1	76.0	89.1
Lifted[35]	87.2	25.3	62.6	80.9	91.2
N-pair [34]	87.9	27.1	66.4	82.9	92.1
Angular[40]	87.8	26.5	**67.9**	**83.2**	92.2
AML[11]	**89.1**	**31.7**	66.3	82.8	**92.5**
OURS	**88.2**	**30.5**	**69.7**	**85.2**	**93.2**

parison. However, our work can be easily combined with any hard negative mining method.

4.3. Evaluation Metrics

Following the standard protocol used in [35, 34], we calculate the Recall@a metric [19] for retrieval task. Specifically, for each query image, top a nearest images will be returned based on Euclidean distance, then the recall score will be 1 if at least one positive image appears in the returned a images and 0 otherwise. For clustering evaluation, we adopt the k-means algorithm to cluster testing instances and the quality is reported in terms of the standard F1 and NMI metrics. Refer to [35] for detailed formulation.

Table 4. Comparison of different boundary α on CUB-200-2011 [39] dataset

Varying α	Recall@a(%)			
	R@1	R@2	R@4	R@8
$\alpha = 26$	52.9	65.4	76.1	85.1
$\alpha = 28$	53.1	65.3	76.1	84.7
$\alpha = 30$	<u>55.1</u>	<u>66.5</u>	<u>76.8</u>	<u>85.3</u>
$\alpha = 32$	54.5	66.0	76.4	85.3

Table 5. Comparison of different margin β on CUB-200-2011 [39] dataset

Varying β	Recall@a(%)			
	R@1	R@2	R@4	R@8
$\beta = 0.1$	51.9	64.5	75.8	84.9
$\beta = 0.3$	52.7	64.9	75.6	84.4
$\beta = 0.5$	<u>55.1</u>	<u>66.5</u>	<u>76.8</u>	<u>85.3</u>
$\beta = 0.7$	53.9	65.7	76.2	85.3
$\beta = 1.0$	53.2	66.3	76.8	85.4

4.4. Result Analysis

Retrieval and clustering. Table 1, 2 and 3 report the clustering and retrieval results for CUB-200-2011, Cars196 and Stanford Online Products separately. We color the best results with red and the second best with blue. The Comparison between traditional contrastive or triplet and Lifted or N-pair shows that hard data mining indeed help to boost the performance. N-pair can be cooperated with many metric learning approaches and achieve improvement mainly because of the advance in its batch construction. Among all baselines, our proposed method DAMLRRM achieves state-of-the-art performance in most cases. It worth mentioning that, DAMLRRM does not need complicated offline data preprocessing and release from hard data mining.

Figure 4 and 5 show the visualization results of CUB-200-2011 and Cars196, which implemented by dimensionality reduction algorithm t-SNE [24]. We zoom in four regions to highlight several representative classes and the various colors of the bounding box are corresponding to different categories. Two of the zoom-in regions are used for demonstrating the compact feature embedding of intra-class and the rest two for illustrating the discrimination between different classes. Despite the large pose and appearance variation, our method effectively generates a significant feature mapping that preserves semantic similarity. Figure 6 gives some instances of query and top-5 ranking images for Stanford Online Products. Despite the huge changes in the viewpoint, configuration, and illumination, our method can successfully retrieve instances from the same class.

Ablation study: effect of boundary α and margin β. There are two hyperparameters involved in our method, which are boundary α and margin β respectively. Table 4 and 5 study the impact of various parameters for the retrieval task on CUB-200-2011 dataset. We set $\beta = 0.5$

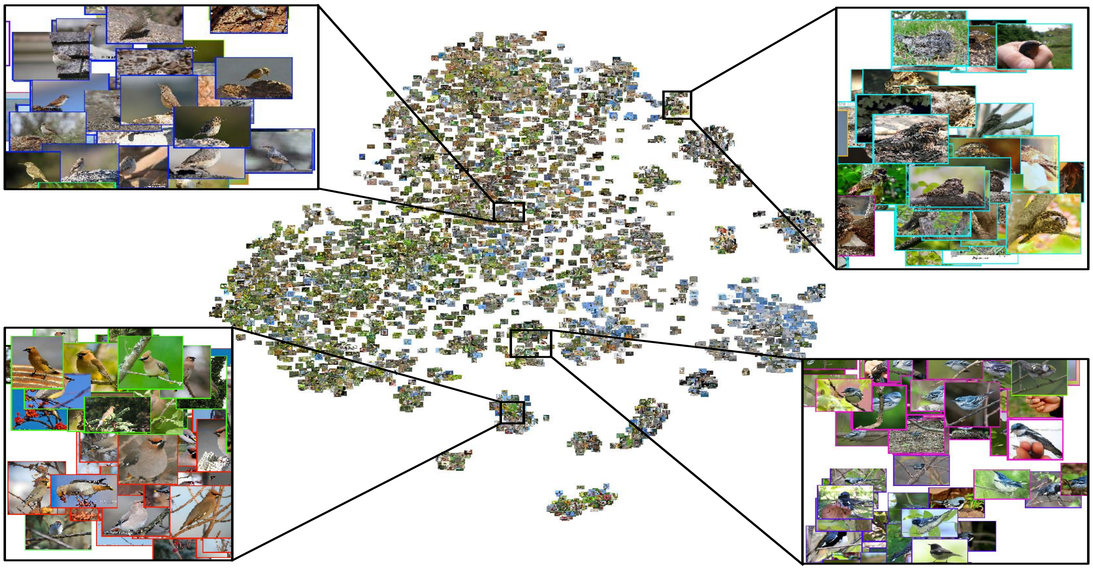


Figure 4. Visualization of feature embedding computed by our method using t-SNE on CUB-200-2011 dataset.

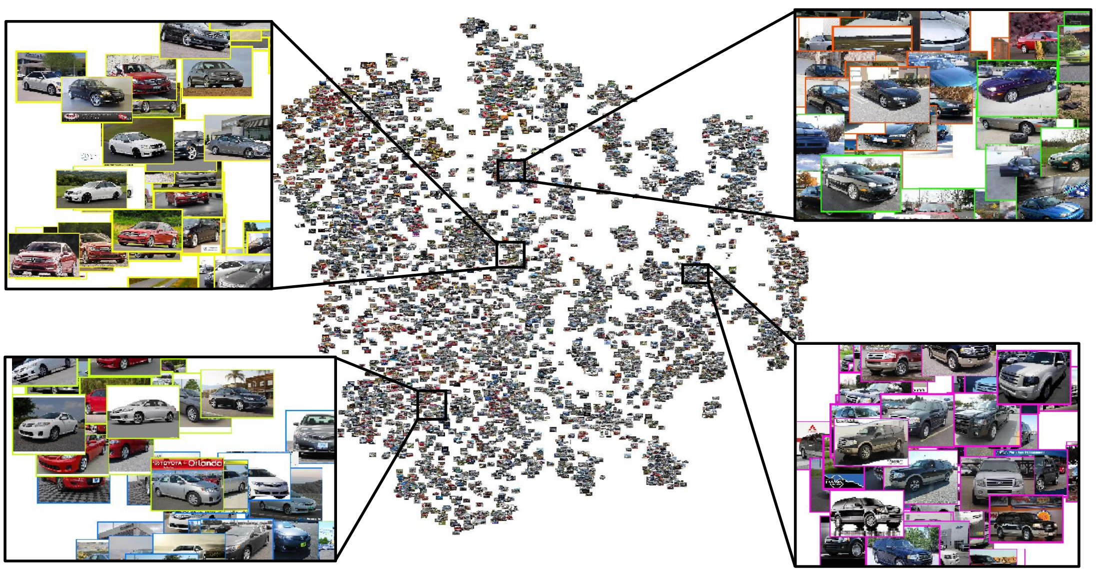


Figure 5. Visualization of feature embedding computed by our method using t-SNE on Cars196 dataset.

Table 6. Comparison of one stream data batch construction and asymmetric data batch construction of CUB-200-2011 [39] dataset

Method	# Sampling Size	Recall@a(%)			
		R@1	R@2	R@4	R@8
Lifted[35]	700K	46.9	59.8	71.2	81.5
N-pair[34]	500K	51.0	63.3	74.3	83.2
OURS[1]	36K	52.3	65.5	76.2	85.5

Table 7. Comparison of full combined positive pairs and *prim* tree connected positive pairs on CUB-200-2011 [39] dataset

Method	Recall@a(%)				
	R@1	R@2	R@4	R@8	R@16
Full PPs	48.2	61.3	72.8	84.3	90.1
OURS [2]	51.2	63.5	74.6	84.5	91.2

when varying the value of α, and set $\alpha = 0.5$ when discussing β. It can be seen that the best performance is obtained when $\alpha = 30, \beta = 0.5$. Furthermore, DAMLRRM is not sensitive to the two parameters and we set boundary to 30 and margin to 0.5 throughout our experiments.

Ablation study: effect of asymmetric batches. In order to verify the effectiveness of asymmetric structure, we

Figure 6. Examples of successful queries on our Stanford Online Products dataset using our embedding (size 512). Images in the first column are query images and the rest are five nearest neighbors.

remove the graph loss and keep the shuffled loss only, which is denoted as OURS^1. We compare it with conventional one stream data batch construction methods: Lifted and N-pair algorithms. Table 6 reports the retrieval metrics of CUB-200-2011 and demonstrates the priority of two asymmetric stream batches construction. Notably, our method only samples about 36K images which are about ten percentage of Lifted and N-pair.

Ablation study: effect of graph pairs construction. To illustrate the difference between two positive training pools established by minimum-cost spanning tree and fully combination, we remove the shuffled loss from DAMLRRM and keep graph loss. We denote them as OURS^2 and Full PPs respectively. Table 7 reports the retrieval result of CUB-200-2011. We can observe that minimum-cost tree based positive pairs training pool is significant for improving the performance, which is mainly because relaxing the constraint employed on positive pairs and the generalization ability is enhanced.

Algorithmic complexity analysis. Compared with Lifted[35] and N-pair[34], our proposed method builds a *prim* tree within each category additionally. The computational complexity of *prim* tree is: $\mathcal{O}_p = \sum_{i=1}^{k-1} i \cdot (k-i)$, where k is the number of instances in a tree. The comparison of training time cost is shown in Table 8, we believe that the additional offline training time is worthy given the significantly improved accuracy. For testing, all instances are mapped by one stream model, and the time cost is the same.

5. Conclusion

In this paper, we propose a novel asymmetric loss for deep metric learning, which targets at mining the rich relationship and enhance generalization ability at the same time.

Table 8. Comparison of training time on CUB_200_2011[39] dataset.

Method	Lifted[35]	N-pair[34]	OURS
Iterations/Sec	2.2	2.2	0.84
Training Time	5.1 h	5.1 h	13.2 h

To min the rich relationship, we construct two structured and quantified asymmetric data streams, which interlace to each other during iterations. Such an asymmetric structure enables continuous newly combined pairs to be compared when optimizing the model, and hence a rich relationship is mined under a small amount of sampling size. To enhance its generalization ability, we relax the constraint on positive pairs. Instead of connecting all possible positive pairs, we build a minimum-cost spanning tree within one category to ensure the form of connected field. Minimum-cost spanning tree based sampling algorithm obeys the inherent distribution of data, where not all positive instances are associated directly. Our proposed model releases from hard data mining and achieves higher accuracy while even at the cost of fewer than ten percents sampling images compared with the peer methods including the lifted method [35] and N-pair [34].

6. Acknowledgment

Our work was also supported by the National Natural Science Foundation of China under Grant 61572388 and 61703327, Key R&D Program-The Key Industry Innovation Chain of Shaanxi under Grant 2017ZDCXL-GY-05-04-02, 2017ZDCXL-GY-05-02, and 2018ZDXM-GY-176, and the National Key R&D Program of China under Grant 2017YFE0104100.

References

[1] 3d object representations for fine-grained categorization. In *International IEEE Workshop on 3D Representation and Recognition*, 2013.

[2] S. Bell and K. Bala. Learning visual similarity for product design with convolutional neural networks. *ACM Transactions on Graphics*, 34(4):98, 2015.

[3] B. Bollobás and O. M. Riordan. Mathematical results on scale-free random graphs. *Handbook of graphs and networks: from the genome to the internet*, pages 1–34, 2003.

[4] J. A. Bondy, U. S. R. Murty, et al. *Graph theory with applications*, volume 290. 1976.

[5] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah. Signature verification using a" siamese" time delay neural network. In *NeurIPS*, pages 737–744, 1994.

[6] M. Bucher, S. Herbin, and F. Jurie. Improving semantic embedding consistency by metric learning for zero-shot classiffication. In *ECCV*, pages 730–746. Springer, 2016.

[7] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, volume 1, pages 539–546, 2005.

[8] Y. Cui, F. Zhou, Y. Lin, and S. Belongie. Fine-grained categorization and dataset bootstrapping using deep metric learning with humans in the loop. In *CVPR*, pages 1153–1162, 2016.

[9] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *ICML*, pages 209–216, 2007.

[10] C. Deng, E. Yang, T. Liu, W. Liu, J. Li, and D. Tao. Unsupervised semantic-preserving adversarial hashing for image search. *IEEE Trans. Image Process.*, 2019.

[11] Y. Duan, W. Zheng, X. Lin, J. Lu, and J. Zhou. Deep adversarial metric learning. In *CVPR*, pages 2780–2789, 2018.

[12] M. Guillaumin, J. Verbeek, and C. Schmid. Is that you? metric learning approaches for face identification. In *ICCV*, pages 498–505, 2009.

[13] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *null*, pages 1735–1742, 2006.

[14] B. Harwood, B. Kumar, G. Carneiro, I. Reid, T. Drummond, et al. Smart mining for deep metric learning. In *ICCV*, pages 2821–2829, 2017.

[15] S. C. Hoi, W. Liu, M. R. Lyu, and W.-Y. Ma. Learning distance metrics with contextual constraints for image retrieval. In *CVPR*, volume 2, pages 2072–2078, 2006.

[16] C. Huang, C. C. Loy, and X. Tang. Local similarity-aware deep feature embedding. In *NeurIPS*, pages 1262–1270, 2016.

[17] N. Ide and K. Suderman. Graf: A graph-based format for linguistic annotations. In *proceedings of the Linguistic Annotation Workshop*, pages 1–8, 2007.

[18] A. Iscen, G. Tolias, Y. Avrithis, and O. Chum. Mining on manifolds: Metric learning without labels. *arXiv preprint arXiv:1803.11095*, 2018.

[19] H. Jegou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *TPAMI*, 33(1):117–128, 2011.

[20] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM International Conference on Multimedia*, pages 675–678, 2014.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[22] C. Li, C. Deng, N. Li, W. Liu, X. Gao, and D. Tao. Self-supervised adversarial hashing networks for cross-modal retrieval. In *CVPR*, pages 4242–4251, 2018.

[23] J. Lu, J. Hu, and Y.-P. Tan. Discriminative deep metric learning for face and kinship verification. *TIP*, 26(9):4269–4282, 2017.

[24] L. V. D. Maaten. *Accelerating t-SNE using tree-based algorithms*. JMLR.org, 2014.

[25] Y. Movshovitz-Attias, A. Toshev, T. K. Leung, S. Ioffe, and S. Singh. No fuss distance metric learning using proxies. In *ICCV*, pages 360–368, 2017.

[26] H. Oh Song, S. Jegelka, V. Rathod, and K. Murphy. Deep metric learning via facility location. In *CVPR*, pages 5382–5390, 2017.

[27] R. C. Prim. Shortest connection networks and some generalizations. *Bell system technical journal*, 36(6):1389–1401, 1957.

[28] O. Rippel, M. Paluri, P. Dollar, and L. Bourdev. Metric learning with adaptive density discrimination. *arXiv preprint arXiv:1511.05939*, 2015.

[29] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, and M. Bernstein. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015.

[30] R. Salakhutdinov and G. Hinton. Learning a nonlinear embedding by preserving class neighbourhood structure. In *Artificial Intelligence and Statistics*, pages 412–419, 2007.

[31] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823, 2015.

[32] F. Shen, Y. Xu, L. Liu, Y. Yang, Z. Huang, and H. T. Shen. Unsupervised deep hashing with similarity-adaptive and discrete optimization. *TPAMI*, 2018.

[33] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[34] K. Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *NeurIPS*, pages 1857–1865. 2016.

[35] H. O. Song, Y. Xiang, S. Jegelka, and S. Savarese. Deep metric learning via lifted structured feature embedding. In *CVPR*, 2016.

[36] Y. Sun, Y. Chen, X. Wang, and X. Tang. Deep learning face representation by joint identification-verification. In *NeurIPS*, pages 1988–1996, 2014.

[37] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. pages 1–9, 2014.

[38] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *CVPR*, pages 1701–1708, 2014.

[39] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, 2011.

[40] J. Wang, F. Zhou, S. Wen, X. Liu, and Y. Lin. Deep metric learning with angular loss. In *ICCV*, pages 2612–2620, 2017.

[41] Y. Wang, J. Choi, V. Morariu, and L. S. Davis. Mining discriminative triplets of patches for fine-grained classification. In *CVPR*, pages 1163–1172, 2016.

[42] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *JMLR*, 10(Feb):207–244, 2009.

[43] C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krähenbühl. Sampling matters in deep embedding learning. In *ICCV*, 2017.

[44] E. P. Xing, M. I. Jordan, S. J. Russell, and A. Y. Ng. Distance metric learning with application to clustering with side-information. In *NeurIPS*, pages 521–528, 2003.

[45] J. Xu, L. Luo, C. Deng, and H. Huang. Bilevel distance metric learning for robust image recognition. In *NeurIPS*, page 42024211, 2018.

[46] J. Xu, L. Luo, C. Deng, and H. Huang. Multi-level metric learning via smoothed wasserstein distance. In *IJCAI*, page 29192925, 2018.

[47] J. Xu, L. Luo, C. Deng, and H. Huang. New robust metric learning model using maximum correntropy criterion. In *SIGKDD*, page 25552564. ACM, 2018.

[48] C. Yang, L. Zhang, H. Lu, X. Ruan, and M.-H. Yang. Saliency detection via graph-based manifold ranking. In *CVPR*, pages 3166–3173, 2013.

[49] E. Yang, C. Deng, C. Li, W. Liu, J. Li, and D. Tao. Shared predictive cross-modal deep quantization. *IEEE Trans. Neural Netw. Learn. Syst.*, 29(11):5292–5303, 2018.

[50] E. Yang, C. Deng, W. Liu, X. Liu, D. Tao, and X. Gao. Pairwise relationship guided deep hashing for cross-modal retrieval. In *AAAI*, pages 1618–1625, 2017.

[51] E. Yang, T. Liu, C. Deng, and D. Tao. Adversarial examples for hamming space search. *IEEE Trans. Cybern.*, 2018.

[52] M. Yang, C. Deng, and F. Nie. Adaptive-weighting discriminative regression for multi-view classification. *Pattern Recogn.*, 88(4):236–245, 2019.

[53] X. Yang, C. Deng, X. Liu, and F. Nie. New l2, 1-norm relaxation of multi-way graph cut for clustering. In *AAAI*, 2018.

[54] Y. Yuan, K. Yang, and C. Zhang. Hard-aware deeply cascaded embedding. In *CVPR*, pages 814–823, 2017.

[55] X. Zhang, F. Zhou, Y. Lin, and S. Zhang. Embedding label structures for fine-grained feature representation. In *CVPR*, pages 1114–1123, 2016.

[56] Z. Zhang and V. Saligrama. Zero-shot learning via joint latent similarity embedding. In *CVPR*, pages 6034–6042, 2016.

[57] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NeurIPS*, pages 321–328, 2004.

[58] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *NeurIPS*, pages 169–176, 2004.