Self-supervised Synthesis Ranking for Deep Metric Learning

Zheren Fu, Zhendong Mao, Chenggang Yan, An-An Liu, *Member, IEEE*, Hongtao Xie, and Yongdong Zhang, *Senior Member, IEEE*

***Abstract*—The core purpose of deep metric learning is to construct an embedding space, where objects belonging to the same class are gathered together and the ones from different classes are pushed apart. Most existing approaches typically insist to inter-class characteristics, *e.g.*, class-level information or instance-level similarity, to obtain semantic relevance of data points and get a large margin between different classes in the embedding space. However, the intra-class characteristics, *e.g.*, local manifold structure or relative relationship within the same class, are usually overlooked in the learning process. Hence the output embeddings have limitation in retrieving a good ranking result if existing multiple positive samples. And the local data structure of embedding space cannot be fully exploited since lack of relative ranking information. As a result, the model is prone to overfitting on a train set and get low generalization on the test set (unseen classes) when losing sight of intra-class variance. This paper presents a novel self-supervised synthesis ranking auxiliary framework, which captures intra-class characteristics as well as inter-class characteristics for better metric learning. Our method designs a synthetic samples generation of polar coordinates to generate measurable intra-class variance with different strength and diversity in the latent space, which can simulate the various local structure change of intra-class in the initial data domain. And then formulates a self-supervised learning procedure to fully exploit this property and preserve it in the embedding space. As a result, the learned embedding space not only keeps inter-class discrimination but also owns subtle intra-class diversity, leading to better global and local embedding structures. Extensive experiments on five benchmarks show that our method significantly improves and outperforms the state-of-the-art methods on the performances of both retrieval and ranking by 2%-4%.**

***Index Terms*—Deep Metric Learning, Image Retrieval, Self-Supervised Learning, Generative Model.**

I. INTRODUCTION

This work is supported in part by the National Natural Science Foundation of China under Grant U19A2057, in part by the Fundamental Research Funds for the Central Universities under Grant WK3480000008 and Grant WK3480000010. Zhendong Mao is the corresponding author.

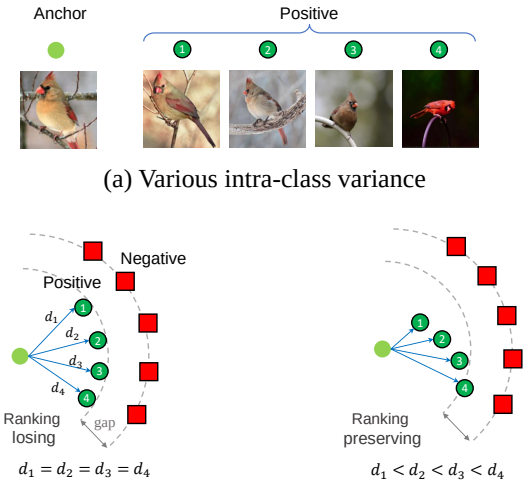
Zhendong Mao and Hongtao Xie are with the School of Information Science and Technology, University of Science and Technology of China, Hefei 230022, China (e-mail: zdmao@ustc.edu.cn; htxie@ustc.edu.cn). Yongdong Zhang is with the School of Information Science and Technology, University of Science and Technology of China and the Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei 230022, China (e-mail: zhyd73@ustc.edu.cn). Zheren Fu is with the School of Cyberspace Science and Technology, University of Science and Technology of China, Hefei 230027, China (e-mail: fzr@mail.ustc.edu.cn).

Chenggang Yan is with Intelligent Information Processing Lab, Hangzhou Dianzi University, Hangzhou 310018, China (e-mail: cgyan@hdu.edu.cn). An-An Liu is with School of Electrical and Information Engineering, Tianjin University, Tianjin 300072, China (e-mail: anan0422@gmail.com).

DEEP metric learning aims to learn effective distance or similarity measures among arbitrary data points through deep neural networks. It defines an embedding space where semantically similar samples (*e.g.*, images of the same class) are close together, and dissimilar ones (*e.g.*, images from different classes) are far apart. Since its powerful representation ability of instances, deep metric learning has been widely applied in a variety of computer vision tasks, including image retrieval [1], [2], person re-identification [3]–[5], visual tracking [6]–[8], face recognition [9], [10] and crowd counting [11].

The paradigm of deep metric learning focuses on devising proper loss functions [10], [12]–[15], which use binary supervision to indicates pairwise distances between an anchor (regarded as a query in retrieval) and its positive or negative samples. Their common target is to minimize the distance of positive pairs and maximize the distance of negative pairs. These losses extremely depend on mining strategy [14], [16], pairs weighting [15], [17], or samples generation [18], [19] to exploit informative and hard samples from mini-batchs. Recent works also use ensemble methods to improve performances, such as attentions [20]–[22], features separation [23]–[25], and reinforcement learning [26]. These boost-like approaches are based on typical metric learning losses to distinguish harder negative pairs. In general, the above methods center on learning more class-discriminative embedding space by maximizing inter-class variance as far as possible.

However, existing deep metric learning approaches entirely disregard intrinsic intra-class variance during embedding learning. Intra-class variance contains the relative distances between anchor points and positive samples, or local manifold structure in the embedding space. They regard all positive samples equally since the lack of annotations and try their best to discriminate positive and negative samples, while the ranking of different positive samples is discarded totally. That's to say previous methods mainly concentrate on how to maximize the inter-class variance and increase the margin of different class-neighbors in the embedding space, while the intra-class variance is minimized and local structure is destroyed unconsciously. Figure 1(a) shows the latent intra-class variance within a certain class, without considering this property, previous methods would learn a less efficient embedding space, shown in Figure 1(b), as compared with the optimal result in Figure 1(c). Besides, if all positive pairs are aligned too equally and closely without any proper constraints [27], [28], it is prone to overfitting on a train set and lack of generalization capability on the test set (unseen classes). In short, previous methods are not able to fully exploit



(a) Various intra-class variance

(b) Previous methods

(c) Our method

Fig. 1. Visualization of the limitation of conventional metric learning. For positive samples retrieval, the best ranking result should be $1, 2, 3, 4$, but previous methods may get a wrong relative order (e.g., $2, 4, 1, 3$), since they overlook the relative relationships of different positive samples in embedding space. (a) Inherent intra-class variances from the same class. (b) Previous methods only maximize the inter-class variance but cannot preserve the intra-class properties. (c) Our method captures intra-class variance by keeping their ranking information to achieve better retrieval and ranking results.

the intra-class variance, which is indispensable for better ranking results, and also very crucial to learn discriminative embeddings and robust model on unseen classes.

In this paper, we propose a novel self-supervised synthesis ranking (SSR) auxiliary framework, which defines a standard criterion to generate and measure intra-class variance, then use a self-supervised learning procedure to preserve their ranking relationships in the embedding space. As shown in Fig. 2, we first present a novel synthetic samples generation of polar coordinates to get quantifiable intra-class variance from real samples. These synthetic samples are generated based on the polar coordinate system of latent hyperspace, where the radial distances mostly represent different semantic strength of intra-class variance, and the directional angles mainly reflect their abundant semantic diversity. Since deep networks can learn high-level representations with semantic abstractions [29], [30], and directions in the feature space correspond to semantic transforms [31], [32], e.g., changing the color or viewpoint of an object. Secondly, we propose the self-supervised ranking preserving method, which derives a unique ranking loss for synthetic samples to exploit their intra-class variance ranking relationships in the embedding space, and also can be easily integrated with existing deep metric learning methods for inter-class variance mining. In this way, the learned embeddings not only maintain inter-class separability but also discriminate subtle intra-class variance, leading to a better global and local embedding structure for retrieval and ranking.

The contributions of this paper are summarized as follows:

- We design a typical paradigm to preserve the local structure of the embedding space by generating and quantifying inherent intra-class variance. To the best of our knowledge, our method is the first self-supervised auxiliary framework to capture both intra-class and inter-class variance for deep metric learning.
- We propose a novel synthetic samples generation of polar coordinates to obtain controllable intra-class variance in the latent space, where their semantic strength and diversity can be measured appropriately, and firstly apply synthetic samples to the self-supervised learning.
- We present a powerful ranking preserving loss function, which can support the model to not only exploit intra-class intrinsic characteristics, but also learn inter-class discriminative semantic embeddings.
- Extensive evaluation experiments on five common benchmarks demonstrate that our method improves and outperforms the performances of state-of-the-art approaches on both retrieval and ranking 2%-4%.

A previous conference version [33] of our work has been accepted in AAAI 2021. Compared with [33], we propose a new generation and measure method of intra-class variance for self-supervised learning, which is the core of our motivation and framework. The previous version just uses simple image transform functions (e.g., Random Crop, Perspective Transform, Color Jitter) to simulate the changes of intra-class variance. These transforms are not only hard to control the semantic strength of intra-class variance in quantity, but also limited by finite image transform types for semantic diversity. Our proposed synthetic sample generation of polar coordinates can strictly and effectively perform intra-class variance with quantifiable semantic strength and diversity, through the latent hyperspace and implicit ways. Besides, our improved method gets better performances than the previous work [33] on three famous benchmarks. We also conduct more experiments on larger datasets and more informative evaluation protocols [34], then further investigate the effects of batch sizes, backbones, and embedding dimensions in ablation studies.

II. RELATED WORK

A. Deep Metric Learning

Deep Metric learning aims to learn a representation space where similar samples are pushed together and dissimilar samples are repelled against, with the advent of deep neural networks. To build the embedding space, plenty of loss functions have been proposed with desired properties and can be categorized into two classes, pair-based and proxy-based.

Pair-based metric losses [13], [14] take pairs of samples to constitute groups of pairwise distances. Typical examples are contrastive loss [35] and triplet loss [36], which take two-tuples and three-tuples samples respectively. Then N-pair loss [12] and Lifted-structured loss [37] exploit multiple samples to mine richer structural information. Recent pair-based losses are proposed by mining strategies [16] and pair weightings [15], [17] to improve the final performances. But these methods lead to a biased model due to unbalanced selection between easy and hard samples [38] during training.

The proxy-based losses [39], [40] propose learnable proxy embeddings the class-related representation and a part of network parameters. They encourage each image as the anchor point to be close to the proxies of the same class and far

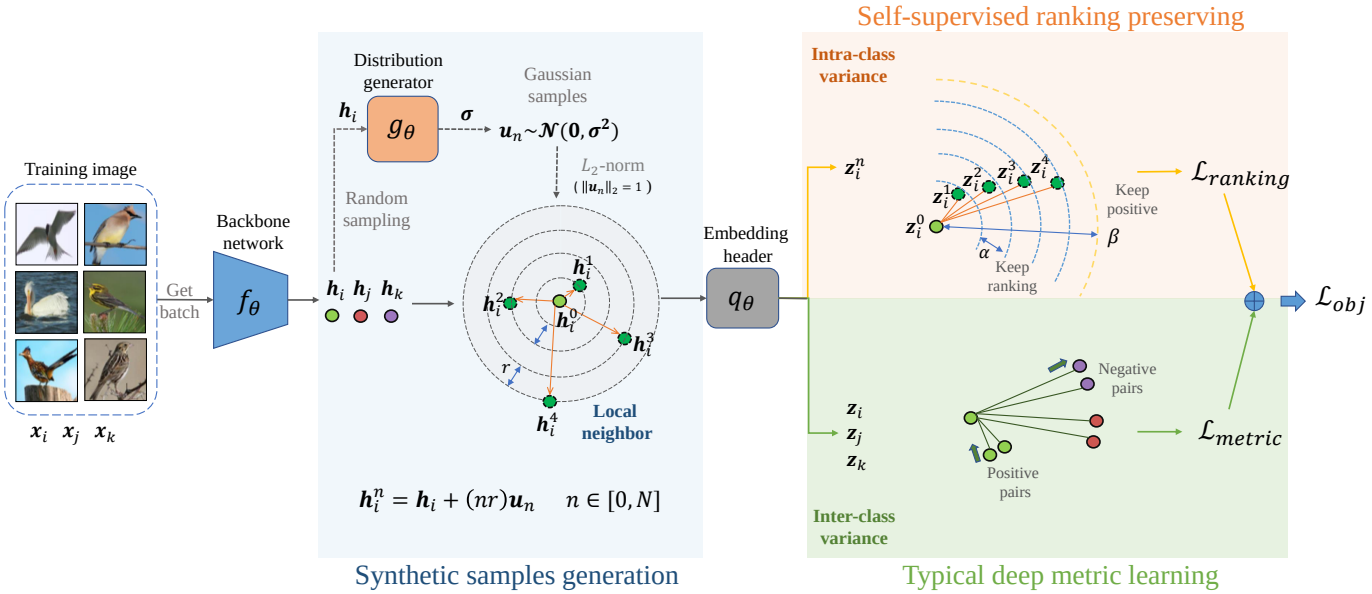


Fig. 2. Overview of our proposed self-supervised synthesis ranking (SSR) auxiliary framework, where any deep metric learning algorithms can be applied for inter-class variance mining (green block). And the SSR consists of two steps: synthetic samples generation (blue block) and self-supervised ranking preserving (orange block). First, the backbone network f_θ get latent features $\boldsymbol{h}_i$ from images $\boldsymbol{x}_i$, the quantifiable intra-class variance is generated by synthetic samples in the latent space. Through the vector linear operation of polar coordinates, we can get multiple synthetic positive samples $\boldsymbol{h}_i^n$ with different strength (radius on polar coordinates) and rich diversity (angle) of intra-class variance to the anchor point $\boldsymbol{h}_i^0$ ($\boldsymbol{h}_i$), depending on n. Then embedding header q_θ maps the latent space to embedding space $\boldsymbol{z}_i$, the self-supervised surrogate loss $\mathcal{L}_{ranking}$ aims to preserving generative intra-class variance and their ranking characteristics on learnt embedding space. Finally, the auxiliary loss is attached to typical deep metric learning loss $\mathcal{L}_{metric}$ to optimize the whole model.

away from these of different classes, instead of other image points. The standard cross-entropy loss for image classification with the final full-connected layer can be regarded as one of metric learning proxy-based loss. These losses reduce the computational complexity and obtain faster convergence when the number of classes is small, otherwise the parameters are tremendous and lead to out of memory [38].

Besides the above losses, ensemble methods [21], [22] are effective to boost the performance. MIC [24] strengthens inter-class discriminative features through characteristics shared across classes. Divide [23] uses the divide-and-conquer algorithm to learn many of partitional embedding spaces. XBM [41] finds slow drift phenomena during embedding training then uses the memory mechanism to expand large batch sizes.

B. Self-supervised learning

Self-supervised learning (SSL) aims to learn discriminative feature representations without relying on manual annotations. It is usually used as a pre-training process for diverse vision downstream tasks, such as classification, detection, and segmentation [42]. The training powers come from a variety of well-designed pretext tasks which can learn inherent attributes of unlabelled data. Early methods included image inpainting [43], and rotation prediction [44]. Recently, contrastive based self-supervised methods [42] have shown strong performance and close to (even stronger than) traditional supervised learning. Contrastive learning tries to decrease the distance between representations of augmented views from the same image (as positive pairs), meanwhile, increase the distance between representations of different augmented views from different images (as negative pairs). Their paradigm is defined on pair-wise relations and similar to pair-based deep metric learning methods. What's more, self-supervised learning is helpful to solve some specific problems [45]. Metric learning also employs its idea to obtain the more discriminative embeddings [24], [41]. In contrast, our framework makes use of the SSL to generate and capture intrinsic intra-class variance.

C. Sample Generation

Recently, sample generation [18], [46], [47] have been proposed to produce potential hard samples for performance boost of deep metric learning. It aims to exploit lots of easy negative samples and train the model with extra sample-pair relationships. For example, Duan et al. [46] and Zhao et al. [48] are the first to use generative adversarial networks to produce adversarial hard samples. Then Zheng et al. [49] and Lin et al. [18] leverage the auto-encoders networks to generative virtual samples and control their hard levels. But the above methods require additional network architectures or adversarial strategy, which can lead to harder optimization, slower training speed, and more redundant parameters [19]. To solve these problems, recent works [47], [50] generate virtual samples or classes, for pair-based and proxy-based losses by simple algebraic computation in the embedding space. However, they just utilize the intuitive generative strategy, which is global and ignores sample-related information.

III. Preliminaries

This section introduces the mathematical formulation of deep metric learning. Let $\boldsymbol{\mathcal{X}} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_K\}$ denotes a dataset

of training images in the RGB domain, and $\mathcal{Y} = \{y_i\}_{i=1}^{K} \in [1, 2, \ldots, C]$ are the corresponding labels. Deep metric learning aims to learn a feature mapping $\boldsymbol{\mathcal{X}} \rightarrow \boldsymbol{\mathcal{Z}}$, which projects original data space $\boldsymbol{x}_i \in \boldsymbol{\mathcal{X}}$ to the embedding space $\boldsymbol{z}_i \in \boldsymbol{\mathcal{Z}}$ by deep neural networks: $\mathbb{R}^{H\times W\times 3} \rightarrow \mathbb{R}^{D}$. The training goal is to learn the model parameters such that embeddings of similar images are close together while dissimilar ones are far apart. Formally, the distance metric between two images $\boldsymbol{x}_i, \boldsymbol{x}_j$ in the embedding space is defined as:

$$d(\boldsymbol{z}_i, \boldsymbol{z}_j) = \|\boldsymbol{z}_i - \boldsymbol{z}_j\|_2 . \tag{1}$$

where $d(,)$ is the Euclidean distance between two vectors. We also can define the similarity metric by computing cosine similarity: $s(\boldsymbol{z}_i, \boldsymbol{z}_j) = \boldsymbol{z}_i \boldsymbol{z}_j^T / (\|\boldsymbol{z}_i\|_2 \|\boldsymbol{z}_j\|_2)$. L2-normalization usually is applied to the embeddings ($\|\boldsymbol{z}_i\|_2 = \|\boldsymbol{z}_j\|_2 = 1$), so two metrics are equivalent.

After taking these metric methods, various kinds of metric learning loss functions $\mathcal{L}_{metric}$ [14], [15], [36] have been proposed in recent years to learn the discriminative embeddings by exploiting the semantic relationship between images.

Triplet loss [36] is the fundamental metric learning loss. It considers three-tuples points and pulls the anchor point $\boldsymbol{z}_a$ closer to the positive point $\boldsymbol{z}_p$ of the same class ($y_a = y_p$) than to the negative point z_n of the different class ($y_a \neq y_n$) by a fixed margin m:

$$\mathcal{L}_{Triplet} = \frac{1}{|\mathcal{T}|} \sum_{(a,p,n)\in\mathcal{T}} [d_{ap} - d_{an} + \alpha]_+, \tag{2}$$

where $d_{ap} = d(\boldsymbol{z}_a, \boldsymbol{z}_p)$, $d_{an} = d(\boldsymbol{z}_a, \boldsymbol{z}_n)$, and $[]_+$ are the hinge function. Triplet samples sets $\mathcal{T}$ are constructed by various sampling strategies [16] from a mini-batch.

Margin loss [14] extends the standard triplet loss by introducing a dynamic and learnable boundary β between positive pairs $\mathcal{P}$ and negative pairs $\mathcal{N}$. It transfers the common triplet ranking problem to a relative ordering of pairs:

$$\mathcal{L}_{Margin} = \gamma + \frac{1}{|\mathcal{P}|} \sum_{(a,p)\in\mathcal{P}} (d_{ap} - \beta) + \frac{1}{|\mathcal{N}|} \sum_{(a,n)\in\mathcal{N}} (\beta - d_{an}) \tag{3}$$

where γ is fixed margin. Margin loss utilize the distance-weighted triplet sampling method to construct sample pairs.

Multi-Similarity loss (MS loss) [15] is one of the latest work for deep metric learning. Unlike triplet based methods, it adds self-similarity and relative similarities of pairs, which mine and weight more informative samples in a mini-batch. Given an anchor $\boldsymbol{z}_i$ point, corresponding positive pairs $\mathcal{P}_i$ and negative pairs $\mathcal{N}_i$ are selected with specific boundary:

$$\mathcal{P}_i = \{s_{ij} | s_{ij} > \min_{y_k = y_i} s_{ik} - \epsilon\} \tag{4}$$

$$\mathcal{N}_i = \{s_{ij} | s_{ij} < \max_{y_k \neq y_i} s_{ik} + \epsilon\} \tag{5}$$

where the ϵ is a fixed threshold and $s_{ij} = s(\boldsymbol{z}_i, \boldsymbol{z}_j)$ (cosine similarity). Then MS loss can be formulated as:

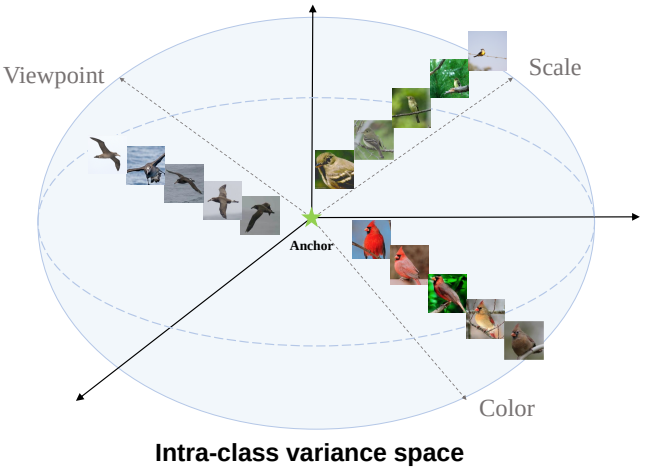


Fig. 3. Visualization of inherent intra-class variance space in original data domain. We show the gradual semantic changes of scale, color, and viewpoint. Examples come from three different classes in CUB-200-2011 dataset [51].

$$\mathcal{L}_{MS} = \frac{1}{|\mathcal{B}|} \sum_{i\in\mathcal{B}} \left[\frac{1}{\alpha} \log \left\{ 1 + \sum_{k\in\mathcal{P}_i} e^{-\alpha(s_{ik} - \lambda)} \right\} + \frac{1}{\beta} \log \left\{ \sum_{k\in\mathcal{N}_i} e^{\beta(s_{ik} - \lambda)} \right\} \right] \tag{6}$$

where α, β, and λ are hyper-parameters, and $\mathcal{B}$ denotes a mini-batch of samples.

IV. APPROACH

In this section, we first give the definition of intra-class variance and learning principle, then introduce our self-supervised synthesis ranking (SSR) framework as shown in Fig. 2, which follows two steps: synthetic samples generation and self-supervised ranking preserving. The first step (blue block) aims to find an approximate intra-class variance generation and measure, which can be simulated on a local neighbor of the latent feature space, because the semantic relations between samples on deep feature spaces can be captured by the relative positions of their features [30]–[32]. Then we get synthetic samples having quantifiable intra-class variance with different semantic strength and diversity. The second step (orange block) constructs a ranking preserving loss for generative samples by self-supervised learning, to keep their ranking relationships on the embedding space, which can be easily integrated with existing deep metric learning methods (green block) and exploit local embedding structures for more discriminative and robust semantic embeddings.

A. Definition of intra-class variance

Intra-class variance is the diverse visual representations of the semantic similar object such as scale, color, viewpoint, and so on. It is fine-grained detail changes under the certain class contrast to inter-class variance. Given an image $\boldsymbol{x}_a$, its positive sample $\boldsymbol{x}_p \in \boldsymbol{\mathcal{X}}_p$ and negative sample $\boldsymbol{x}_n \in \boldsymbol{\mathcal{X}}_n$ ($y_a = y_p \neq y_n$) to learn representations $\boldsymbol{z}_a, \boldsymbol{z}_p, \boldsymbol{z}_n$ in the

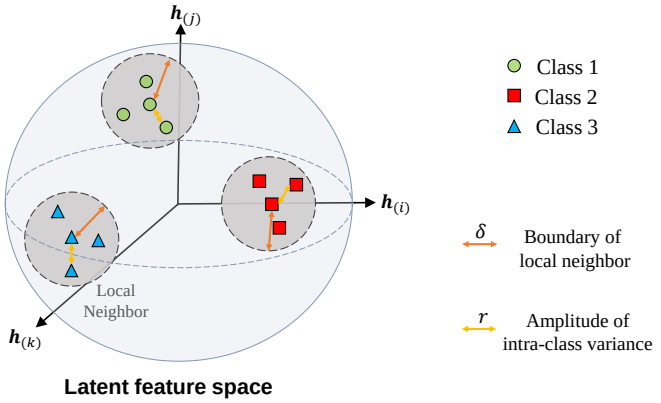


Fig. 4. Illustration of latent feature space $\boldsymbol{h}_i$ with well-defined local neighbor and measurable intra-class variance when given an anchor point.

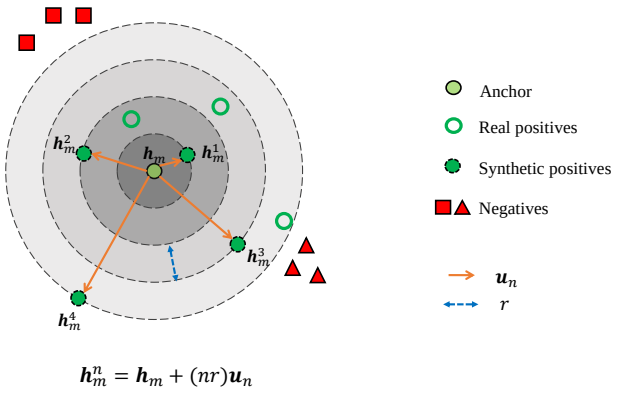


Fig. 5. Illustration of our synthetic samples generation of polar coordinates. We get multiple synthetic samples with measurable semantic strength and diversity of intra-class variance on the local neighbor of latent feature space.

embedding space. The intra-class and inter-class variances are defined as $d(\boldsymbol{z}_a, \boldsymbol{z}_p)$ and $d(\boldsymbol{z}_a, \boldsymbol{z}_n)$ respectively. The fundamental metric learning methods focus on increasing the margin between them, so as to ensure the following constraint hold:

$$\max_{\boldsymbol{x}_p \in \mathcal{X}_p} d(\boldsymbol{z}_a, \boldsymbol{z}_p) < \min_{\boldsymbol{x}_n \in \mathcal{X}_n} d(\boldsymbol{z}_a, \boldsymbol{z}_n). \tag{7}$$

The metric losses on Section III are typical. thus only inter-class variance (margin between difference classes) is optimized, while the intrinsic intra-class variance (margin within the same class) is ignored. In order to get more robust and generalized metric learning model on unseen classes, we need to keep the intra-class variance properly. For a image $\boldsymbol{x}_a$ with its positives $\boldsymbol{x}_{p1}, \boldsymbol{x}_{p2}$ ($y_a = y_{p1} = y_{p2}$), it is desirable that the following constraint also hold [52]:

$$\text{if } d_M(\boldsymbol{x}_a, \boldsymbol{x}_{p1}) < d_M(\boldsymbol{x}_a, \boldsymbol{x}_{p2}), \text{ then } d(\boldsymbol{z}_a, \boldsymbol{z}_{p1}) < d(\boldsymbol{z}_a, \boldsymbol{z}_{p2}). \tag{8}$$

where d_M is the measure operator of intra-class variance space in the original data domain (image RGB space), as shown in Figure 3. Eq (8) is our learning principle and means the relationships between intra-class variance, *e.g.*, relative rankings, in the embedding space are consistent with those in the original image domain. Current human-labeled signal, *e.g.*, class label or pairwise label, treats images from the same category equally, *i.e.*, two images are similar, it not able to further distinguish between similar images. It's significant to find a proper metric d_M to quantify intra-class variance, we start with the help of self-supervised learning.

B. Synthetic samples generation

One of the acceptable metrics d_M is based on the fact that high-level representations learned by deep convolutional networks can potentially capture abstractions with meaningful semantics [29], [30]. Specifically, translating deep features along various directions has been shown to be corresponding to performing different semantic transformations on the input images, e.g., the color or viewpoint changes of an object with the same class [31]. Recent work on semantic augmentation [32] reveal that we can learn all kinds of semantic directions and controllable strength in the deep feature space, which can also be leveraged to perform semantic intra-class variance efficiently. Therefore, translating real samples along proper gaps and directions in the latent space can get measurable intra-class variance with various strength and diversity.

Generally, the whole mapping models of deep metric learning are comprised of two parts: a representation network f_θ and an embedding header q_θ with related parameters θ. Therefore, we can get two-stage features for images $\boldsymbol{x}_i$,

$$\boldsymbol{h}_i = f_\theta(\boldsymbol{x}_i) \in \mathbb{R}^F, \quad \boldsymbol{z}_i = q_\theta(\boldsymbol{h}_i) \in \mathbb{R}^D \tag{9}$$

f_θ usually is a Convolutional Neural Network with global pooling and regarded as backbone networks. q_θ generally is a fully-connected layer to finish the mapping from latent feature space $\boldsymbol{h}_i$ to the final embedding space $\boldsymbol{z}_i$. As aforementioned, intra-class variance of images $\boldsymbol{x}_i$ can be represented on the Euclidean ball of latent space, called local neighbor:

$$\mathbf{N}(\boldsymbol{h}_i) = \{\boldsymbol{h}_k \mid y_k = y_i, \|\boldsymbol{h}_k - \boldsymbol{h}_i\|_2 \le \delta\} \tag{10}$$

δ is a threshold controlling the region size of $\mathbf{N}(\boldsymbol{h}_i)$ using Euclidean distance $d(\boldsymbol{h}_k, \boldsymbol{h}_i)$, to ensure that $\boldsymbol{h}_k$ belong to the same class of $\boldsymbol{h}_i$ as rational intra-class variance. The specific threshold δ can be estimated with existing training dataset $\boldsymbol{\mathcal{X}}$ and representation network f_θ. Besides, on a polar coordinate system of latent hyperspaces, the difference between $\boldsymbol{h}_k$ and $\boldsymbol{h}_i$ can be decomposed into two parts: the radial distance (aka radius) r and the directional angle (aka angle) $\boldsymbol{u}$.

$$\begin{aligned} \boldsymbol{h}_k - \boldsymbol{h}_i &= \|\boldsymbol{h_k} - \boldsymbol{h_i}\|_2 \frac{\boldsymbol{h_k} - \boldsymbol{h_i}}{\|\boldsymbol{h_k} - \boldsymbol{h_i}\|_2} \\ &= r\boldsymbol{u}, \ \boldsymbol{h}_k \in \mathbf{N}(\boldsymbol{h}_i) \end{aligned} \tag{11}$$

According to [31], [32], the semantic relations between samples can be captured by the relative positions in latent feature space. And the radius r (scalar) mainly represents the semantic strength of intra-class variance and the angle $\boldsymbol{u}$ (unit vector) mostly reflects the semantic diversity of intra-class variance, which is also important but ignored by previous sample generation methods [19], [47]. We represent the relationship between $\boldsymbol{h}_k$ and $\boldsymbol{h}_i$ through the polar coordinate properties of latent space (r_k and $\boldsymbol{u}_k$):

$$\boldsymbol{h}_k = \boldsymbol{h}_i + r_k \boldsymbol{u}_k \quad r_k \le \delta, \|\boldsymbol{u}_k\|_2 = 1 \tag{12}$$

TABLE I
SUMMARY OF IMPORTANT NOTATIONS AND INTERPRETATION.

Not.	Interpretation	Not.	Interpretation
$d(,)$	Euclidean distance	$\boldsymbol{x}_i$	original data
$s(,)$	cosine similarity	$\boldsymbol{z}_i$	embeddings
$d_M(,)$	measure of intra-class variance	y_i	class label
$f_\theta()$	backbone network	$\boldsymbol{h}_i$	latent feature
$q_\theta()$	a fully-connected layer	δ	threshold of $\mathbf{N}$
$\mathbf{N}()$	local neighbor	$\boldsymbol{h}_m^n$	synthetic latent feature
$g_\theta()$	distribution generator	r	strength of variance
$\boldsymbol{\sigma}$	distribution parameters	$\boldsymbol{u}$	diversity of variance
m	index of original samples	n	index of synthetic samples
M	batch size of original samples	N	number of synthetic samples
α	ranking margin	$\boldsymbol{z}_m^n$	synthetic samples
τ	scale factor	β	positive boundary

Since the intra-class variance space of $\boldsymbol{h}_i$ is consistent to the local neighbor $\mathbf{N}(\boldsymbol{h}_i)$ in the latent space, the measure operator of intra-clas variance d_M is available on $\mathbf{N}(\boldsymbol{h}_i)$ and we hold:

$$\begin{aligned} &\text{if } r_{k1} < r_{k2}, \\ &\text{then } d(\boldsymbol{h}_i, \boldsymbol{h}_{k1}) < d(\boldsymbol{h}_i, \boldsymbol{h}_{k2}), \\ &\text{and then } d_M(\boldsymbol{x}_i, \boldsymbol{x}_{k1}) < d_M(\boldsymbol{x}_i, \boldsymbol{x}_{k2}). \end{aligned} \tag{13}$$

Based on Eq. (12) and Eq. (13), our synthetic sample generation is shown in Fig. 5 and Eq. (14). In a training mini-batch, we denote the m-th image as $\boldsymbol{x}_m$ ($m = 1, 2, ..., M$), and its latent feature $\boldsymbol{h_m}$, then we generate multiple synthetic samples with measurable intra-class variance from two perspective, the semantic strengths nr (r is a fixed scalar, n is a positive integer) and semantic directions $\boldsymbol{u}_n$.

$$\boldsymbol{h}_m^n = \boldsymbol{h}_m + (nr)\boldsymbol{u}_n, \ n = 0, 1, ..., N \ \ (\boldsymbol{h}_m^0 = \boldsymbol{h}_m) \tag{14}$$

Specially, $\boldsymbol{u}_n$ are sampled from the sample-conditional zero-mean Gaussian distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{\sigma}^2)$, then are L2-normalized to ensure $\|\boldsymbol{u}_n\| = 1$. We use a distribution generator g_θ to learn the diagonal covariance matrices, $\boldsymbol{\sigma} = g_\theta(\boldsymbol{h}_m)$. To promote the generator g_θ to learn more meaningful semantic directions for diverse intra-class variance, we use the KL-divergence loss (D_{KL}) to constrain it close to the standard Gaussian distribution as the regularization term [18].

$$\begin{aligned} \mathcal{L}_{dist} &= D_{KL}\left[\mathcal{N}(\mathbf{0}, \boldsymbol{\sigma}^2)||\mathcal{N}(\mathbf{0}, \mathbf{I})\right] \\ &= \frac{1}{2}\sum_{i=1}^{F}\left[\sigma_{(i)}^2 - \log\sigma_{(i)}^2 - 1\right] \end{aligned} \tag{15}$$

Alternatively, we can estimate the class-conditional covariance matrices of the latent features for each class, and then constrain our learned diagonal covariance matrices close to the estimated matrices. But most metric learning datasets only have a few samples for each class, the estimation is difficult and can result in trivial solutions easily. So we still adopt the regularization way to keep our learned covariance matrices within reasonable bounds, like Eq. (15).

C. Self-supervised ranking preserving

In order to keep the ranking relationships of intra-class variance on the embedding space as the principle Eq. (8), we derive a powerful ranking preserving loss function and then exploit the local embedding structure. We use the head encoder q_θ to get the embeddings $\boldsymbol{z}_m^n = q_\theta(\boldsymbol{h}_m^n)$ of synthetic samples $\boldsymbol{h}_m^n$. According to the self-supervised strategy and Eq. (13), the embedding of the original image $\boldsymbol{x}_m$, $\boldsymbol{z}_m^0$, should be closer to the embeddings of synthetic samples with lower intra-class variance strength (smaller n) than higher strength (larger n):

$$\text{if } i < j, \text{ then } d(\boldsymbol{z}_m^0, \boldsymbol{z}_m^i) < d(\boldsymbol{z}_m^0, \boldsymbol{z}_m^j). \tag{16}$$

This ranking preserving objective is formulated based on the pairwise ranking loss, i.e., triplet loss [36]. Without loss of generality, we use cosine similarity $s(,)$ rather $d(,)$ in our following introduction since all embeddings are L2-normalized. The similarity of an embedding pair with weaker differences of intra-class variance strength should be larger than that with stronger ones in the embedding space by a fixed margin α.

$$\mathcal{L}_{base} = \left[s(\boldsymbol{z}_m^0, \boldsymbol{z}_m^j) - s(\boldsymbol{z}_m^0, \boldsymbol{z}_m^i) + \alpha\right]_+, \text{when } i < j. \tag{17}$$

Then we use listwise ranking to integrate all synthetic samples in mini-batch. it's made up of the sequential Eq. (17):

$$\mathcal{L}_{list} = \frac{1}{M}\sum_{m=1}^{M}\sum_{n=1}^{N-1}[s(\boldsymbol{z}_m^0, \boldsymbol{z}_m^{n+1}) - s(\boldsymbol{z}_m^0, \boldsymbol{z}_m^n) + \alpha]_+, \tag{18}$$

Besides, we apply the LogSumExp and SoftPlus functions [15], [53] to smooth Eq. (18). After summing over all of them, the loss function becomes:

$$\mathcal{L}_{sort} = \frac{1}{M}\sum_{m=1}^{M}\frac{1}{\tau}log[1 + \sum_{n=1}^{N-1}e^{\tau(-\mathcal{S}_{m,n} + \mathcal{S}_{m,n+1} + \alpha)}]. \tag{19}$$

where $\mathcal{S}_{m,n}$ is $s(\boldsymbol{z}_m^0, \boldsymbol{z}_m^n)$ and τ is the scale factor.

Eq. (18) has a limitation that gradients are fixed, the value is ± 1 when a training pair violates the constraint and 0 otherwise. The loss cannot mine any informative sample pairs [15] and leads to the trivial samples [17] during training. By contrast, the derivative of Eq. (19) is weighted according to the relative hardness, which is the degree of strength that a pair violates the constraint. As shown in Eq. (20), a harder pair can get larger gradient magnitudes.

$$\frac{\partial\mathcal{L}_{sort}}{\partial\mathcal{S}_{m,n}} = \frac{e^{\tau(-\mathcal{S}_{m,n-1} + \mathcal{S}_{m,n} + \alpha)} - e^{\tau(-\mathcal{S}_{m,n} + \mathcal{S}_{m,n+1} + \alpha)}}{1 + \sum_{j=1}^{N-1}e^{\tau(-\mathcal{S}_{m,j} + \mathcal{S}_{m,j+1} + \alpha)}}. \tag{20}$$

Although $\boldsymbol{h}_m^n$ is generated by certain transformations, it is still in the local neighbor $\mathbf{N}(\boldsymbol{h}_i)$ and a positive sample for $\boldsymbol{x}_m$. So we add the constraint of positive pairs to ensure $\mathcal{S}_{m,n}$ is larger than a boundary β. We also use the smooth version of the pointwise loss [53] to control the relative hardness:

$$\mathcal{L}_{pos} = \frac{1}{M}\sum_{m=1}^{M}\frac{1}{\tau}log[1 + \sum_{n=1}^{N}e^{-\tau(\mathcal{S}_{m,n} - \beta)}]. \tag{21}$$

The derivative of Eq. (21) is the same situation, the harder positive pairs with a lower similarity are assigned with a larger weight, then get larger gradient to be optimized.

$$\frac{\partial\mathcal{L}_{pos}}{\partial\mathcal{S}_{m,n}} = -\frac{e^{-\tau(\mathcal{S}_{m,n} - \beta)}}{1 + \sum_{n=1}^{N}e^{-\tau(\mathcal{S}_{m,n} - \beta)}}. \tag{22}$$

Algorithm 1 Model training process with our method

Input:
images X, class labels Y,
neural networks $f_\theta, g_\theta, q_\theta$,
hyper-parameters $\alpha, \beta, \tau, r, \lambda, p_{task}$
Output:
network parameters θ $(\theta_f, \theta_g, \theta_q)$

$epoch \leftarrow 0$
while Not Converged **do**
  repeat
    $x, y \leftarrow MiniBatch(X, Y)$
    $h \leftarrow Latent(x; f_\theta)$
    $z \leftarrow Embedding(h; q_\theta)$
    $Compute\ \mathcal{L}_{metric}(z, y)$
    if $p < p_{task}, p \sim U(0,1)$ **then**
      $\boldsymbol{\sigma} \leftarrow g_\theta(h)$
      $u_n \leftarrow L_2Norm\left[u_n \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\sigma}^2)\right]$
      $h^n \leftarrow h + (rn)(u_n)$
      $z^n \leftarrow Embedding(h^n; q_\theta)$
      $Compute\ \mathcal{L}_{ranking}(z^n)$
    end if
    $Compute\ \mathcal{L}_{obj} = \mathcal{L}_{metric} + \lambda\mathcal{L}_{ranking}$
    $\theta \leftarrow Backward(\mathcal{L}_{obj})$
  until Epoch End
  $epoch \leftarrow epoch + 1$
end while

With the weighted sum of Eq. (19), Eq. (21), and Eq. (15), we reach the self-supervised list-wise ranking loss:

$$\mathcal{L}_{ranking} = \mathcal{L}_{sort} + \mathcal{L}_{pos} + \mathcal{L}_{dist}, \tag{23}$$

Our auxiliary framework is independent of the choice of metric learning losses $\mathcal{L}_{metric}$, which is explained in Section III. We just incorporate incorporate $\mathcal{L}_{ranking}$ into $\mathcal{L}_{metric}$ and train the entire networks.

$$\mathcal{L}_{obj} = \mathcal{L}_{metric} + \lambda\mathcal{L}_{ranking}. \tag{24}$$

The overall objective $\mathcal{L}_{obj}$ makes up of a general metric learning loss and our proposed self-supervised ranking learning loss, where λ weights the importance of intra-class variance. The whole training procedure is outlined in Alg. 1.

V. Analysis

A. Comparison to Existing Works

Many existing metric learning methods have used the synthetic samples generation to boost the performance [19], [46]–[49]. However, both our generation strategy and the motivation fundamentally differ from previous methods. We aims to preserve ranking characteristics of intra-class variance by generating synthetic samples, whose semantic strength are controllable and measurable. In contrast, previous methods use synthetic samples to discriminate inter-class variance further, and they don not focus on the strength changes of synthetic samples and their ranking relationship.

These are also other multi-task and self-supervised auxiliary methods for deep metric learning, like MIC [24] and DiVA [25]. Although these methods all focus on modeling intra-class variation, we are distinct from MIC and DiVA in many aspects. They mine inter-class or class-shared information, which are separated from class-specific properties. In comparison, we preserve the ranking characteristics of intra-class variance for every class to explore local structure of embedding space. What's more, we don't use extra network units, such as gradient reversal layers, which will bring performance instability and parameter complexity during training.

Compared with previous version [33], we propose a novel generation and measure method of intra-class variance for self-supervised learning, which is the core of our motivation and framework. The previous work just uses simple image transforms and augmentations to simulate the changes of intra-class variance. These transforms are not only hard to control the semantic strength of intra-class variance in quantity, but also limited by finite image transform types for semantic diversity. Our proposed synthetic sample generation of polar coordinates can strictly and effectively perform intra-class variance with quantifiable semantic strength and diversity, through the latent hyperspace and implicit ways.

B. Loss complexity

Our proposed ranking loss, $\mathcal{L}_{ranking}$, belongs to general pair-based metric loss, which usually has high computational complexity with large batches. When mini-batch size is M, $O(M^2)$ is for Contrastive loss [35] and Triplet loss [36] with samples mining strategies, $O(M^3)$ is for Lifted-Structure loss [37] and N-pair loss [12]. Margin loss [14] and MS loss [15] also have the $O(M^2)$ complexity. In contrast, the complexity of $\mathcal{L}_{ranking}$ is only $O(M)$, when the number of synthetic samples is limited. Low time consumption and fast convergence speed are advantageous for training.

C. Computation cost

Although the model process the metric learning task and self-supervised task jointly, the latter only requires tiny forward computation and network parameters on the end of framework. Even though the number of synthetic samples increases, the additional runtime and memory cost are negligible for the training stage. (see Section VI-F4 for more details) It is also worth noting that our method does not increase any runtime and parameter for the inference stage.

D. Model Robustness

Since the synthetic samples are sampled from a Gaussian distribution, and virtual compared with real samples from the datasets. They might be noisy training signals. To sovle this problem, we learn the distribution parameters with reasonable constraints ($\mathcal{L}_{dist}$), control the change of semantic strength (r) strictly, and construct proper boundary between original and synthetic samples ($\mathcal{L}_{pos}$). Therefore, these reliable synthetic samples can alleviate the mismatch between the generated distribution and the ground truth distribution. (see Section VI-H and Section VI-G for more explanations and analysis).

TABLE II
RETRIEVAL PERFORMANCES ON FOUR STANDARD BENCHMARK AND THREE BASELINES, WHERE 'SR' REFERS TO THE PREVIOUS VERSION [33]. WE LIST OTHER METRIC LEARNING METHODS USING SAMPLE GENERATION. ALL OF THEM USE THE GOOGLENET [54] AS BACKBONE AND 512 AS EMBEDDING SIZE. $ReImp$ INDICATES OUR RE-IMPLEMENTATION WITH OFFICIAL SETTINGS. OUR TRIPLET USES THE SEMIHARD NEGATIVE MINING [37].

	CUB-200-2011			Cars-196			Stanford Online Products			In-shop Clothes Retrieval		
Method	R@1	R@2	R@4	R@1	R@2	R@4	R@1	R@10	R@100	R@1	R@10	R@20
Triplet+DAML [48]	37.6	49.3	61.3	60.6	72.5	82.5	58.1	75.0	88.0	-	-	-
Triplet+HDML [49]	43.6	55.8	67.7	61.0	72.6	80.7	58.5	75.5	88.3	-	-	-
Triplet+DVML [18]	43.7	56.0	67.8	64.3	73.7	79.2	66.5	82.3	91.8	-	-	-
Triplet+Symm [19]	55.0	67.3	77.5	69.7	78.7	86.1	68.5	82.4	91.3	-	-	-
Triplet+EE [47]	51.7	63.5	74.5	71.6	80.7	87.5	77.2	89.6	95.5	-	-	-
MS+EE [19]	57.4	68.7	79.5	76.1	84.2	89.8	78.1	90.3	95.8	-	-	-
Triplet [36] ($ReImp$)	53.4	65.0	75.4	67.0	77.5	85.2	73.2	87.6	95.0	84.1	95.2	96.5
Triplet+SR [33]	55.0	67.0	77.1	70.1	79.3	86.0	75.0	88.6	95.5	85.5	96.6	97.7
Triplet+SSR	**56.1**	**67.3**	**77.6**	**71.4**	**80.2**	**86.8**	**75.4**	**88.8**	**95.6**	**86.3**	**96.9**	**97.9**
Margin [14] ($ReImp$)	55.4	67.5	77.7	76.4	84.7	89.8	73.7	87.5	94.1	83.8	95.5	97.0
Margin+SR [33]	57.3	69.0	78.8	79.4	86.3	91.0	74.9	88.1	95.0	85.4	96.2	97.3
Marin+SSR	**58.3**	**70.0**	**79.3**	**80.5**	**87.7**	**91.9**	**75.8**	**88.7**	**95.2**	**86.2**	**96.6**	**97.5**
MS [15] ($ReImp$)	56.2	68.3	79.1	77.0	84.3	89.9	75.3	89.0	95.4	84.0	95.8	97.2
MS+SR [33]	57.4	69.3	79.8	80.9	88.2	92.6	76.5	89.6	95.9	85.5	96.6	97.8
MS+SSR	**58.2**	**70.6**	**81.1**	**82.0**	**88.6**	**93.3**	**76.8**	**89.8**	**96.0**	**86.6**	**97.2**	**98.2**

VI. EXPERIMENTS

A. Datasets

We evaluate our proposed method on five widely-used datasets by following the standard protocol [37] of train and test set split. The first two of these are small datasets and have plenty of positive samples in each class. The last three are large datasets and have a few positive samples.

(1) **CUB-200-2011 (CUB)** [51] contains 11,788 images of 200 species of birds. We use 5,864 images of its first 100 classes for training and 5,924 images of the remaining classes for testing. (2) **Cars-196 (CARS)** [55] contains 16,185 images of 196 car models. We use 8,054 images of its first 98 classes for training and 8,131 images of the other classes for testing. (3) **Stanford Online Products (SOP)** [37] contains 120,053 online product images of 22,634 categories sold on eBay.com. We use 59,551 images of 11,318 classes for training and 60,502 images of the rest classes for testing. (4) **In-shop Clothes Retrieval (InShop)** [56] contains 72,712 clothing images of 7,986 categories. We use 25,882 images of the first 3,997 classes for training and 28,760 images of the other classes for testing. And the test set is further divided into a query set and a gallery set, with 14,218 images of 3,985 classes and 12,612 images of 3,985 classes respectively. (5) **PKU VehicleID (Vehicle)** [57] contains 221,736 images of 26,267 vehicles categories captured by surveillance cameras. we use 110,178 images of 13,134 classes for training and 111,585 images of the other classes for testing. We evaluate on the predefined small, medium and large test sets which contain 800, 1,600 and 2,400 classes respectively.

B. Implementation Details

1) Networks: We implement our method on the GPU of NVIDIA RTX 2080Ti or 3090. For a fair comparison, we use GoogleNet [54], Inception with batch normalization [58], and ResNet50 [59] as the backbone networks. We replace its last layer with a randomly initialized fully-connected layer q_θ for metric learning. An MLP with a 512-dim hidden layer g_θ is attached to learn parameters of the Gaussian distribution. The output embeddings are L_2 normalized before computing distances, and the dimensions are 128 or 512.

2) Optimization: The input images are first resized to 256 $\times$ 256, then cropped to 224 $\times$ 224. For training, we use random crop and random horizontal flips for data augmentation. For testing, we only use the single center crop. We use Adam optimizer with $4e^{-4}$ weight decay. The initial learning rate is 10^{-4} and scaled up 10 times on the output layers for faster convergence. Mini-batches are constructed with the balanced sampler (P classes, K samples per class).

3) Hyperparameter: We set $\alpha = 0.05, \beta = 0.5, \tau = 12, p_{task} = 0.6, \lambda = 0.15, M = 24, N = 5$ for all experiments as [33]. Especially, the new hyper-parameter $r = [1.0, 5.0]$.

C. Results

We evaluate our framework base on three typical deep metric learning losses, which are introduced in Section III: Triplet loss [36], Margin loss [14], MS loss [15]. The parameters are consistent with [33]. We implement them with GoogLeNet and 512 embedding size with 120 batch sizes for fairness. We also list other synthetic sample generation based methods with the same backbone and embedding size for a fair comparison.

First, we evaluate the retrieval performance in terms of the common retrieval metric, the Recall@K (R@K). Tab. II show that our method brings considerable improvement on all baselines and benchmarks. It proves that our method helps models learn discriminative inter-class variance since local structures of the embedding space are mined sufficiently, and it is stimulative to learn class-related boundaries. Three metric learning losses obtain surprising promotions with our framework, 1%-4% R@1 gains on all datasets. In brief, our model is a universal auxiliary framework for deep metric learning regardless of object categories and loss functions. Compared with the previous version work (SR [33]), our method still gets obvious performance improvement.

Qualitative retrieval results are shown in Fig. 6. Our method promotes models to learn more robust embeddings by capturing intra-class variance. Now the new embeddings can

TABLE III
COMPARISON WITH THE STATE-OF-THE-ART DEEP METRIC LEARNING METHODS. BACKBONE NETWORKS OF THE MODELS ARE DENOTED BY ABBREVIATIONS: G–GOOGLENET [54], BN–INCEPTIONBN [58], R–RESNET50 [59]. SUPERSCRIPTS IN THE NETWORKS DENOTE EMBEDDING SIZES. † INDICATES THAT USES THE DISTANCE-BASED SAMPLING AS MARGIN LOSS [14].

		CUB-200-2011				Cars-196				Stanford Online Products			
Method	Setting	R@1	R@2	R@4	R@8	R@1	R@2	R@4	R@8	R@1	R@10	R@100	R@1000
A-BIER [21]	G^{512}	57.5	68.7	78.3	86.2	82.0	89.0	93.2	96.1	74.2	86.9	94.0	97.8
ABE [22]	G^{512}	60.6	71.5	79.8	87.4	85.2	90.5	94.0	96.1	76.3	88.4	94.8	98.2
HTL [13]	BN^{512}	57.1	68.8	78.7	86.5	81.4	88.0	92.7	95.7	74.8	88.3	94.8	98.4
RLL-H [17]	BN^{512}	57.4	69.7	79.2	86.9	74.0	83.6	90.1	94.1	76.1	89.1	95.4	-
SoftTriple [40]	BN^{512}	65.4	76.4	84.5	90.4	84.5	90.7	94.5	96.9	78.3	90.3	95.9	-
Circle [10]	BN^{512}	66.7	77.4	86.2	91.2	83.4	89.8	94.1	96.5	78.3	90.5	96.1	98.6
XBM [41]	BN^{512}	65.8	75.9	84.0	89.9	82.0	88.7	93.1	96.1	**79.5**	90.8	96.1	98.7
ProxyAnchor [38]	BN^{512}	68.4	79.2	86.8	91.6	86.1	91.7	95.0	97.3	79.1	90.8	96.2	98.7
MIC [24]	R^{128}	66.1	76.8	85.6	-	82.6	89.1	93.2	-	77.2	89.4	95.6	-
Divide [23]	R^{128}	65.9	76.6	84.4	90.6	84.6	90.7	94.1	96.5	75.9	88.4	94.9	98.1
PADS [26]	R^{128}	67.3	78.0	85.9	-	83.5	89.7	93.8	-	76.5	89.0	95.4	-
RaMBO [60]	R^{512}	63.5	74.8	84.1	90.4	-	-	-	-	77.8	90.1	95.9	98.7
Margin+SR [33]	R^{128}	66.5	76.8	85.5	91.0	84.5	90.2	93.7	96.1	77.9	89.5	95.4	98.4
Margin+SR [33]	R^{512}	68.2	78.1	86.5	**91.6**	87.7	92.5	95.4	97.3	78.6	90.6	96.2	98.7
Triplet† +SSR	BN^{512}	65.4	76.3	84.3	90.3	80.1	87.3	92.2	95.1	78.9	**91.0**	**96.2**	**98.8**
Margin+SSR	R^{128}	66.3	77.2	85.5	91.3	83.7	89.5	93.3	95.9	78.2	90.5	96.0	98.6
Margin+SSR	R^{512}	**69.1**	**78.8**	**86.6**	91.4	**88.0**	**92.7**	**95.7**	**97.4**	79.1	90.7	96.2	98.7

TABLE IV
COMPARISON WITH THE STATE-OF-THE-ART METHODS ON IN-SHOP CLOTHES RETRIEVAL DATASETS. † INDICATES THAT TRIPLET LOSS USES THE DISTANCE-BASED SAMPLING STRATEGY AS MARGIN LOSS [14].

		In-shop Clothes Retrieval			
Method	Setting	R@1	R@10	R@20	R@30
A-BIER [21]	G^{512}	83.1	95.1	96.9	97.8
ABE [22]	G^{512}	87.3	96.7	97.9	98.5
MS [15]	BN^{128}	88.0	97.2	98.1	98.5
ProxyAnchor [38]	BN^{128}	90.8	97.9	98.5	**99.0**
HTL [13]	BN^{512}	80.9	94.3	95.8	97.2
XBM [41]	BN^{512}	89.9	97.6	98.4	98.6
MIC [24]	R^{128}	88.2	97.0	-	98.0
Divide [23]	R^{128}	85.7	95.5	96.9	97.5
MS+SR [33]	R^{128}	87.8	97.2	98.0	98.5
Margin+SR [33]	R^{128}	88.0	97.3	98.2	98.6
Triplet†+SSR	BN^{512}	**91.0**	**98.0**	**98.7**	**99.0**
Margin+SSR	R^{128}	88.6	97.4	98.3	98.7
Margin+SSR	R^{512}	90.4	97.8	98.6	99.0

TABLE V
COMPARISON WITH THE STATE-OF-THE-ART METHODS ON PKU VEHICLEID DATASET. THE TEST SETS ARE SPLITED INTO THREE SIZES (THE SMALL, MEDIUM, AND LARGE).

		Small		Medium		Large	
Method	Setting	R@1	R@5	R@1	R@5	R@1	R@5
BIER [20]	G^{512}	82.6	90.6	79.3	88.3	76.0	86.4
A-BIER [21]	G^{512}	86.3	92.7	83.3	88.7	81.9	88.7
MS [15]	BN^{512}	91.0	96.1	89.4	94.8	86.7	93.8
MIC [24]	R^{128}	86.9	93.4	-	-	82.0	91.0
Divide [23]	R^{128}	87.7	92.9	85.7	90.4	82.9	90.2
FastAP [61]	R^{512}	91.9	96.8	90.6	**95.9**	87.5	95.1
MS ($ReImp$)	R^{128}	91.4	96.3	89.5	95.1	86.6	94.1
Margin ($ReImp$)	R^{128}	91.8	96.7	90.3	95.4	87.4	94.4
MS+SR [33]	R^{128}	92.0	96.9	89.8	95.3	87.3	94.4
Margin+SR [33]	R^{128}	92.9	97.8	91.1	95.5	88.6	94.8
MS+SSR	R^{128}	92.8	96.8	91.4	95.6	89.0	95.1
Margin+SSR	R^{128}	**93.5**	**96.9**	**92.0**	**95.9**	**89.7**	**95.3**

help to retrieve images correctly and also keep their relative rankings while the original baseline fails, since there are misleading poses, viewpoints, background, or colors from various semantically similar objects of the same class.

D. Comparison with SOTAs

We compare our approach with the state-of-the-art deep metric learning methods. We list the performances with corresponding settings since the backbone and embedding dimension (generally, larger is better) can affect performances greatly. Tab. III, IV, and V demonstrate that our method outperforms state-of-the-art methods on five typical benchmarks. For example, it surpasses the SOTA deep metric learning losses, such as HTL [13], RLL-H [17], Circle [10], and so on by at least 2% R@1 gains for all the datasets, especially for the current strong model (e.g., SoftTriple [40], ProxyAnchor [38], and XBM [41]). It proves the effectiveness of SSR.

When compared with other boost-like methods, such as MIC [24], Divide [23], PADS [26], and RaMBO [60], with the same backbone and baseline loss (R^{128} + Margin loss), our approach still gets better promotion and generalization. For example, we achieve a higher R@1 performance than Divide [23] by 65.9% $\rightarrow$ 66.3% on CUB, 85.7% $\rightarrow$ 88.6% on Inshop. Our method outperforms MIC [24] by 82.6% $\rightarrow$ 83.7% on CARS, PADS [26] by 76.5% $\rightarrow$ 78.2% on SOP, and all these ensemble methods with a large margin on VehicleID. Besides, our method achieves better performance than the previous conference version work [33] in terms of all the benchmarks and baselines, expect for the R^{128} setting on Cars-196 (There are about 1% R@1 decreasing gaps, probably the latent space can not be learned and measured well in this instance).

It is worth noting that, despite our method uses 128-d embeddings, it still gets better results than some state-of-the-art ensemble methods with 512-d embeddings, such as BIER [20], A-BIER [21], ABE [22] and RaMBO [60]. These results show our method can lead to stronger generalization and construct more discriminative embedding spaces.

TABLE VI
INFORMATIVE EVALUATION PROTOCOLS [34] AND RANKING PERFORMANCES ON FOUR BENCHMARKS. ALL OF THE METHODS USE RESNET50 [59] AS BACKBONE AND 128 AS EMBEDDING SIZE (R^{128}). WE REPORT THE MEAN AND STANDARD DEVIATION OF RESULTS OVER 4 INDEPENDENT RUNS

Method	CUB-200-201		Cars-196		Stanford Online Products		In-shop Clothes Retrieval	
	MAP@R	R-Precision	MAP@R	R-Precision	MAP@R	R-Precision	MAP@R	R-Precision
Contrastive [35]	$23.5_{\pm 0.3}$	$34.5_{\pm 0.3}$	$23.6_{\pm 0.4}$	$34.0_{\pm 0.4}$	$38.3_{\pm 0.2}$	$41.7_{\pm 0.2}$	$44.5_{\pm 0.5}$	$47.5_{\pm 0.5}$
N-pair [12]	$21.7_{\pm 0.5}$	$32.6_{\pm 0.5}$	$20.5_{\pm 0.6}$	$31.9_{\pm 0.6}$	$34.6_{\pm 0.4}$	$38.2_{\pm 0.4}$	$43.7_{\pm 0.6}$	$47.3_{\pm 0.6}$
ProxyNCA [39]	$23.8_{\pm 0.2}$	$34.7_{\pm 0.2}$	$22.9_{\pm 0.2}$	$33.1_{\pm 0.2}$	$38.7_{\pm 0.3}$	$42.2_{\pm 0.3}$	$48.4_{\pm 0.2}$	$52.1_{\pm 0.2}$
Triplet [36]	$22.0_{\pm 0.4}$	$33.1_{\pm 0.4}$	$20.3_{\pm 0.5}$	$31.5_{\pm 0.5}$	$37.0_{\pm 0.3}$	$40.5_{\pm 0.3}$	$44.8_{\pm 0.4}$	$48.5_{\pm 0.4}$
Margin [14]	$23.5_{\pm 0.3}$	$34.6_{\pm 0.3}$	$24.4_{\pm 0.2}$	$34.8_{\pm 0.2}$	$40.8_{\pm 0.1}$	$44.2_{\pm 0.1}$	$50.5_{\pm 0.3}$	$53.4_{\pm 0.3}$
MS [15]	$23.2_{\pm 0.2}$	$34.4_{\pm 0.2}$	$25.2_{\pm 0.3}$	$35.6_{\pm 0.3}$	$40.9_{\pm 0.1}$	$44.3_{\pm 0.1}$	$52.1_{\pm 0.2}$	$54.6_{\pm 0.2}$
Margin+SR [33]	$24.0_{\pm 0.3}$	$34.9_{\pm 0.3}$	$27.5_{\pm 0.2}$	$37.3_{\pm 0.2}$	$41.3_{\pm 0.1}$	$44.5_{\pm 0.1}$	$53.3_{\pm 0.1}$	$56.6_{\pm 0.1}$
MS+SR [33]	$23.5_{\pm 0.3}$	$34.4_{\pm 0.3}$	$27.8_{\pm 0.2}$	$37.8_{\pm 0.2}$	$41.5_{\pm 0.1}$	$44.8_{\pm 0.1}$	$52.8_{\pm 0.1}$	$56.0_{\pm 0.1}$
Margin+SSR	$\mathbf{24.3}_{\pm 0.2}$	$\mathbf{35.2}_{\pm 0.2}$	$27.2_{\pm 0.3}$	$36.9_{\pm 0.3}$	$\mathbf{41.7}_{\pm 0.1}$	$\mathbf{45.0}_{\pm 0.1}$	$\mathbf{53.7}_{\pm 0.1}$	$\mathbf{57.1}_{\pm 0.1}$
MS+SSR	$23.7_{\pm 0.2}$	$34.7_{\pm 0.2}$	$\mathbf{27.9}_{\pm 0.2}$	$\mathbf{37.8}_{\pm 0.2}$	$41.5_{\pm 0.1}$	$44.8_{\pm 0.1}$	$53.0_{\pm 0.2}$	$56.3_{\pm 0.2}$

Fig. 6. Qualitative retrieval results with or without our method on Margin loss [14]. (a), (b), (c), (d), (e) is for five datasets (CUB, CARS, SOP, Inshop, and VehicleID) respectively. For each query image (leftmost with blue edge), the top-5 recall results are presented with left-to-right ranking by relative distances. Correct recall are highlighted with green, while incorrect red.

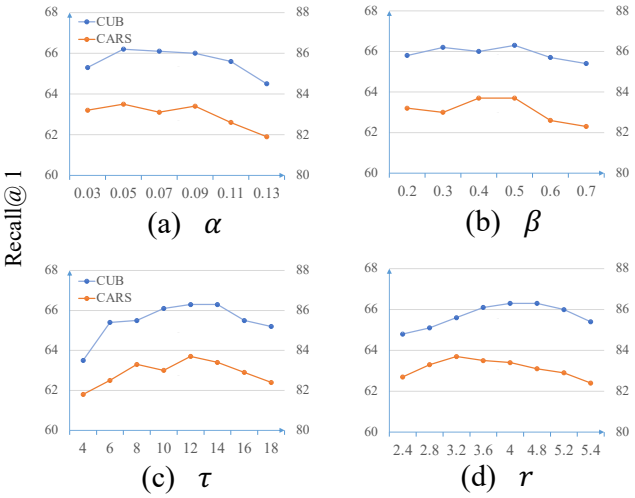


Fig. 7. Recall@1 comparison with various values of four significant hyper-parameters: (a) ranking margin α (in $\mathcal{L}_{sort}$), (b) positive boundary β (in $\mathcal{L}_{pos}$), (c) scale factor τ (in $\mathcal{L}_{sort}, \mathcal{L}_{pos}$), and (d) synthetic radius r.

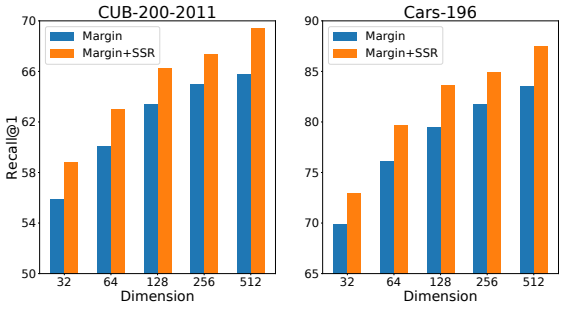


Fig. 8. Recall@1 comparison on various embedding dimensions (32 - 512) on CUB-200-2011 and Cars-196 dataset.

E. Informative Evaluation Protocol

Recent works present more objective evaluation procedures with regard to fairness [34]. Therefore, we evaluate the performance by computing more informative metrics for ranking results, the MAP@R and R-precision. Then we conduct 4 independent runs for each experiment and report the mean and the standard deviation of them for an unbiased evaluation. Tab. VI shows that our method enhances the specific evaluation results for every dataset and new metric.

The MAP@R metric is also reasonable to measure ranking performance and shows our method improves the corresponding results when compared to others, which confirms the effectiveness in learning intra-class variance. There are lots of samples in every class on CUB [51] and CARS [60] datasets, hence their intra-class variances are abundant and the ranking improvements are impressive. However, SOP [37] and InShop [56] datasets has a few examples under each class, the capture of intra-class features is not sensitive for retrieval, which leads to little gain with our method.

F. Ablation Study

We provide ablation experiments to verify the effectiveness of our method and evaluate the contribution of different modules. We choose Margin loss [14] and train models on

TABLE VII
RECALL@1 COMPARISON WITH DIFFERENT MODULES IN THE FRAMEWORK. '✓' MEANS RETAINING THE CORRESPONDING PARTS ON OUR FRAMEWORK OTHERWISE REMOVING. g_θ IS A MULTI-LAYER PERCEPTION IN SR [33].

$\mathcal{L}_{sort}$	$\mathcal{L}_{pos}$	$\mathcal{L}_{dist}$ / g_θ	CUB		CARS	
			SSR	SR	SSR	SR
			63.4		79.5	
✓			64.5	61.9	82.4	78.3
	✓		64.0	60.7	81.3	76.5
	✓	✓	65.3	62.5	82.7	79.4
✓		✓	64.6	64.3	82.1	82.3
✓	✓		65.8	65.8	83.2	84.0
✓	✓	✓	**66.3**	**66.5**	**83.7**	**84.5**

CUB-200-2011 [51] (CUB) and Cars-196 [55] (CARS). We also run multiple times independently and report the mean for each experiment. ResNet50 backbone with 128 embedding sizes is the default setting unless otherwise noted.

1) Hyper-parameters: We show the impact of four important hyper-parameters in Fig. 7: (a) ranking margin α (in $\mathcal{L}_{sort}$), (b) positive boundary β (in $\mathcal{L}_{pos}$), (c) scale factor τ (in $\mathcal{L}_{sort}, \mathcal{L}_{pos}$), and (d) synthetic radius r. When one is variable, the other is fixed as the default setting for controlled experiments. As the boundary of positive pairs, β can't be too large or small otherwise the performances drop heavily. And the performances are stable when α is changed in a proper range, but useless if two small. The scale factor is sensitive to results since it controls the weights of hard synthetic samples. Finally, the size of radius r depends on the threshold δ of the local neighbor in given datasets, which can be estimated by original data distributions and intra-class variance diversity. Note that the setting Section VI is not best since we did not tune them elaborately according to the test set performance.

2) Framework components: In order to analyze the effectiveness of different parts in our proposed ranking loss $\mathcal{L}_{ranking}$, including $\mathcal{L}_{sort}$, $\mathcal{L}_{pos}$ and $\mathcal{L}_{dist}$. We evaluate our framework with different compositions of them. Tab. VII shows that these modules are complementary. When only $\mathcal{L}_{sort}$ or $\mathcal{L}_{pos}$ is incorporated into the self-supervised learning procedure, the performances are inconspicuous. By contrast, the combination help to learn more robust embeddings, and the best result comes with the incorporation of all modules. $\mathcal{L}_{sort}$ is most important and $\mathcal{L}_{pos}, \mathcal{L}_{dist}$ can further enhance the positive influence. We also find that SSR is more effective and robust than SR with different changes of ranking loss.

3) Batch sizes of generative samples: According to previous works [15], [41], [61], large batch sizes can boost the performance of metric loss significantly. To investigate the effect of batch sizes for generative samples, we vary different batch sizes for our ranking loss in Tab. VIII. We find our ranking loss need not large batch sizes and to promote the best performance, unlike typical metric loss functions.

4) Computation cost: Though our model requires an extra self-supervised task, the additional computing time (6% more) and memory cost (1% more) are trivial compared to the baseline, as shown in Tab. VIII. So the memory or computing cost problems [15], [41] are nonexistent during training.

TABLE VIII
RECALL@1 COMPARISON WITH VARIOUS BATCH SIZES ($M \times N$) FOR SYNTHETIC SAMPLES. M IS THE NUMBER OF REAL ANCHOR SAMPLES FOR SYNTHETIC GENERATION AND N IS THE NUMBER OF SYNTHETIC SAMPLES FOR RANKING PRESERVING. WE ALSO REPORT RUNTIME (100 ITERATIONS) AND GPU MEMORY COST FOR SELF-SUPERVISED TASK.

M	N	Batch	CUB	CARS	Runtime	GPU Mem.
0	0	0	63.4	79.5	45.7 s	10.72 GB
16	5	80	65.2	82.7	48.2 s	10.78 GB
20	5	100	65.6	83.0	48.5 s	10.79 GB
28	5	140	65.9	83.3	48.8 s	10.80 GB
32	5	160	64.3	82.8	48.8 s	10.80 GB
24	3	72	64.9	82.6	47.3 s	10.75 GB
24	4	96	65.1	83.5	48.6 s	10.79 GB
24	6	144	65.3	82.8	49.0 s	10.80 GB
24	7	168	64.0	82.6	49.1 s	10.81 GB
24	5	120	**66.3**	**83.7**	48.7 s	10.80 GB

TABLE IX
RECALL@1 COMPARISON WITH DIFFERENT IMAGE DEFORMATIONS IN CARS DATASET. WE CHOOSE THE SAME DEFORMATIONS AS [50].

Image Deformations	Margin	Margin + SSR
Not apply	79.5	83.7 (**+4.2**)
Cutout	70.1	73.5 (**+3.4**)
Dropout	56.5	59.8 (**+3.3**)
Zoom in	59.2	61.3 (**+2.1**)
Zoom out	74.3	76.8 (**+2.5**)
Rotation	67.5	69.4 (**+1.9**)
Shearing	65.2	67.8 (**+2.6**)
Gaussian noise	61.2	63.6 (**+2.4**)
Gaussian blur	69.6	71.1 (**+1.5**)

5) Embedding dimension: In metric learning and similarity search, the trade-off between speed and accuracy is an important issue, where embedded size is the key factor. We test our method with embedding dimensions varying from 32 to 512. The result is quantified in Fig. 8. Our method significantly improves the performance of the baseline in all embedding dimensions. It indicates that our method constructs a highly efficient embedding space for all the dimensionality.

G. Robustness analysis

1) Image deformation: To further evaluate the robustness of embeddings learned by our models, we perform the image deformation test as [50]. As shown in Tab. IX, we add certain deformations which are not used in training to the test images. The results show our models have great robustness and generalization with a large variety of image deformations.

2) Convergence analysis: To evaluate the stability and convergence of ranking loss $\mathcal{L}_{ranking}$, we provide the convergence analysis in the learning process. First, as shown in Fig. 9, our method gets larger metric learning loss $\mathcal{L}_{metric}$ values during train, but higher performances than the baseline on test set, which proves that our method can lead to less overfitting and stronger generalization for models. Besides, the ranking loss can steadily decline and gradually converge with $\mathcal{L}_{metric}$ during learning process, as shown in Fig. 10. It demonstrates that intra-class variance has been preserved well during train, and is helpful to metric learning.

3) Synthetic certainty: Since synthetic samples are sampled from the Gaussian distribution, the quality of generated

TABLE X
RECALL@1 OF SYNTHETIC SAMPLES. WE USE ALL ORIGINAL SAMPLES ON TEST SET AS GALLERY, AND USE SYNTHETIC SAMPLES AS QUERY TO COMPUTE R@1 (FIND THE ORIGINAL SAMPLES WITH THE SAME CLASS, LINE 2). WE ALSO LIST THE GENERAL R@1 PERFORMANCES WITH ORIGINAL SAMPLES AS QUERY (LINE 1).

	Gallery side	
Query side	CUB	CARS
Original (General)	66.3	83.7
Synthetic	**78.5**	**90.4**

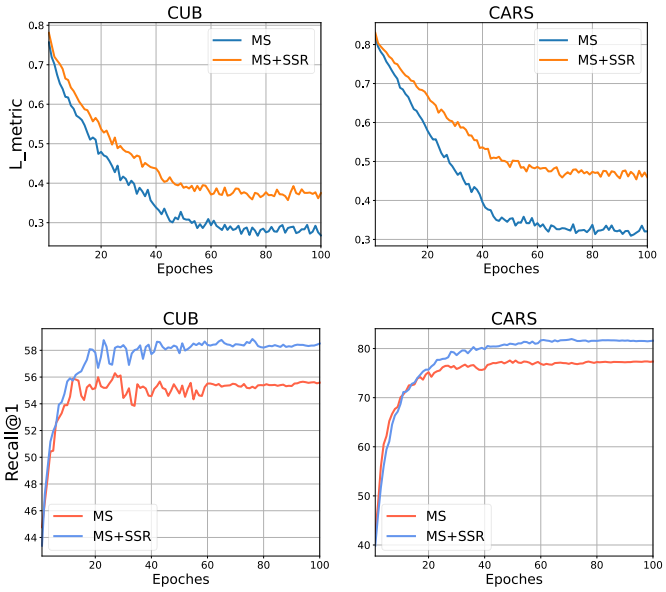


Fig. 9. Metric learning loss curves ($\mathcal{L}_{metric}$) of train set, and Recall@1 curve of test set in the learning process. We train models with GoogleNet backbone and MS loss.

samples have an effect on the performance. We use the synthetic samples as the query side, and compute the Recall@1 performances with the original samples as the gallery side. Tab X shows synthetic samples have higher R@1 performances than original samples, which proves generated samples have high certainty and quality to the original samples and classes.

4) Similarity distribution: To evaluate our method on the effect of similarity distribution, we compute the histograms of cosine similarity of the positive and negative pairs by the models trained with or without our method on test set. The Fig. 11 shows that the similarity distribution of the whole datasets have been improved with our method, since negative pairs and positive pairs are separated further.

H. Visualization

1) Embedding space: To better qualitatively evaluate the embedding space, we illustrate the t-SNE [62] visualizations of image embedding representations by our method on two datasets. As shown in Fig. 13 and Fig. 14, the synthetic samples are generated in-between train samples and can overlap areas of the real training set. Thus, training process can guarantee the authenticity of generated synthetic samples.

2) Pixel domains: To demonstrate that our method generate meaningful semantically synthetic samples, we utilize GAN

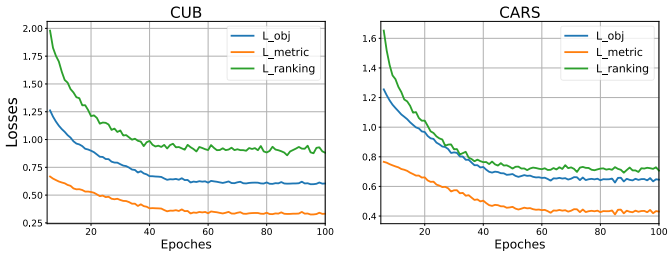


Fig. 10. Different loss curves, which includes $\mathcal{L}_{obj}$, $\mathcal{L}_{metric}$, $\mathcal{L}_{ranking}$ in Eq. (24). $\mathcal{L}_{ranking}$ is the proposed ranking preserving loss.

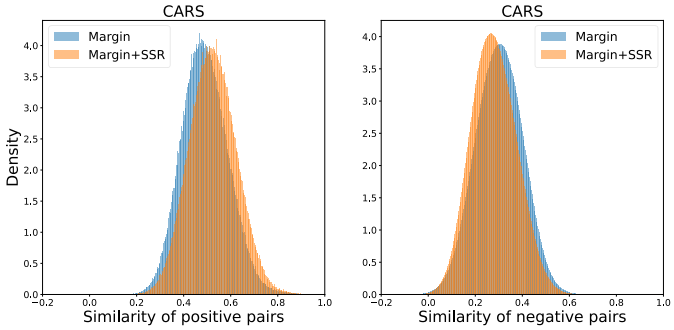


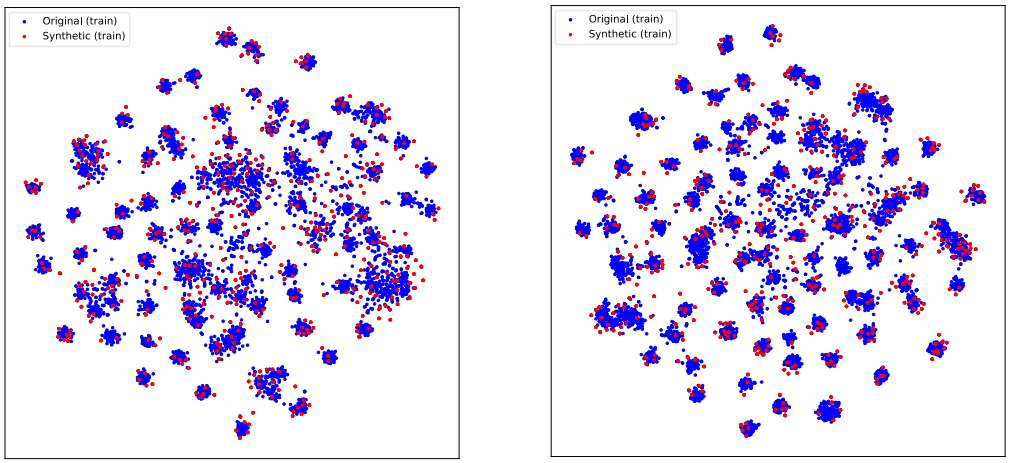
Fig. 11. Histogram of cosine similarities of positive pairs and negative pairs on test set of CARS dataset. We take the whole sample-pairs in the dataset.

[31] to map the synthetic samples back to the pixel space to explicitly show the change of intra-class variance. The visualization results in Fig. 12 show we can get various of virtual samples with different changes of semantic strengths and directions, e.g., viewpoint, background, and color, which are controllable with our generation strategy.

VII. CONCLUSION

This work presents a novel self-supervised synthesis ranking auxiliary framework for deep metric learning. We define the

Fig. 12. Visualization results of our synthetic samples in the pixel space. The first columns represent the original (real) images of four different classes in Cars196. The rest columns present the related synthetic (virtual) images according to original samples by our generation method.



(a) CUB (b) CARS

Fig. 13. t-SNE visualization of embedding representations learnt by our model on CUB-200-2011 and Cars-196 dataset. We show these points of original samples (blue) and generated synthetic samples (red) from the train set.

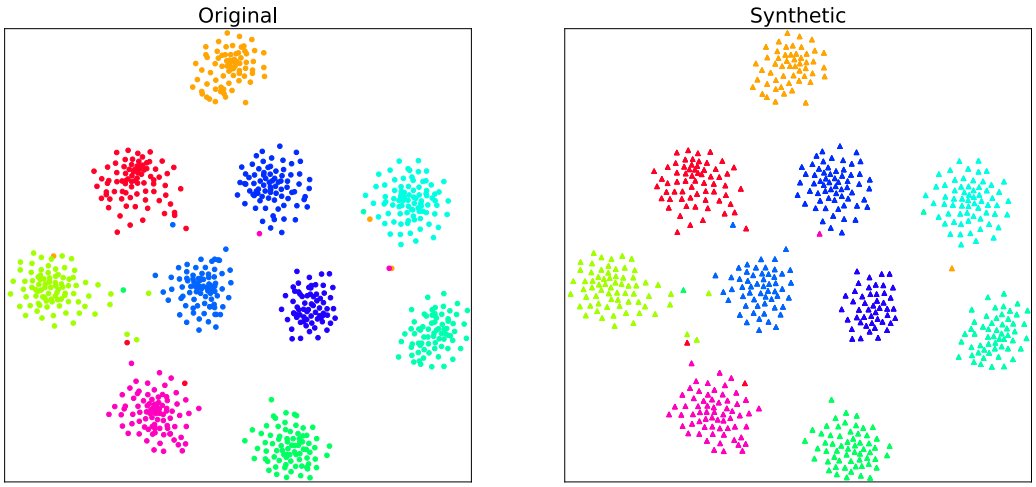


Fig. 14. t-SNE visualization of the embedding representations by our model on Cars-196 dataset. We randomly select 10 classes from the train set. Different colors represent different classes. Then we show original samples from these 10 classes and generated synthetic samples.

standard form of intra-class variance and present a synthetic samples generation strategy to quantify them. A specific ranking preserving auxiliary loss is proposed to maintain their local structure and relative ranking information in the embedding space, which can help models learn more class-discriminative embeddings. Extensive experiments show that our approach significantly improves all baselines, and outperforms state-of-the-art methods on all retrieval and ranking benchmarks.

REFERENCES

[1] W. Jiang, K. Huang, J. Geng, and X. Deng, "Multi-scale metric learning for few-shot learning," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 3, pp. 1091–1102, 2021.

[2] Y. Duan, J. Lu, J. Feng, and J. Zhou, "Deep localized metric learning," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 2644–2656, 2018.

[3] H.-M. Hu, W. Fang, B. Li, and Q. Tian, "An adaptive multi-projection metric learning for person re-identification across non-overlapping cameras," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 9, pp. 2809–2821, 2019.

[4] D. Tao, L. Jin, Y. Wang, Y. Yuan, and X. Li, "Person re-identification by regularized smoothing kiss metric learning," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 10, pp. 1675–1685, 2013.

[5] S. Bak and P. Carr, "Deep deformable patch metric learning for person re-identification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 2690–2702, 2018.

[6] Y. Wu, B. Ma, M. Yang, J. Zhang, and Y. Jia, "Metric learning based structural appearance model for robust visual tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 5, pp. 865–877, 2014.

[7] J. Hu, J. Lu, and Y.-P. Tan, "Deep metric learning for visual tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 11, pp. 2056–2068, 2016.

[8] Y. Cong, B. Fan, J. Liu, J. Luo, and H. Yu, "Speeded up low-rank online metric learning for object tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 6, pp. 922–934, 2015.

[9] J. Lu, G. Wang, and P. Moulin, "Localized multifeature metric learning for image-set-based face recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 3, pp. 529–540, 2016.

[10] Y. Sun, C. Cheng, Y. Zhang, C. Zhang, L. Zheng, Z. Wang, and Y. Wei, "Circle loss: A unified perspective of pair similarity optimization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6398–6407.

[11] Q. Wang, J. Wan, and Y. Yuan, "Deep metric learning for crowdedness regression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 2633–2643, 2018.

[12] K. Sohn, "Improved deep metric learning with multi-class n-pair loss objective," in *Advances in neural information processing systems*, 2016, pp. 1857–1865.

[13] W. Ge, "Deep metric learning with hierarchical triplet loss," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 269–285.

[14] C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krahenbuhl, "Sampling matters in deep embedding learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2840–2848.

[15] X. Wang, X. Han, W. Huang, D. Dong, and M. R. Scott, "Multi-similarity loss with general pair weighting for deep metric learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5022–5030.

[16] Y. Suh, B. Han, W. Kim, and K. M. Lee, "Stochastic class-based hard example mining for deep metric learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7251–7259.

[17] X. Wang, Y. Hua, E. Kodirov, G. Hu, R. Garnier, and N. M. Robertson, "Ranked list loss for deep metric learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5207–5216.

[18] X. Lin, Y. Duan, Q. Dong, J. Lu, and J. Zhou, "Deep variational metric learning," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 689–704.

[19] G. Gu and B. Ko, "Symmetrical synthesis for deep metric learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 10 853–10 860.

[20] M. Opitz, G. Waltner, H. Possegger, and H. Bischof, "Bier-boosting independent embeddings robustly," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5189–5198.

[21] ——, "Deep metric learning with bier: Boosting independent embeddings robustly," *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[22] W. Kim, B. Goyal, K. Chawla, J. Lee, and K. Kwon, "Attention-based ensemble for deep metric learning," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 736–751.

[23] A. Sanakoyeu, V. Tschernezki, U. Buchler, and B. Ommer, "Divide and conquer the embedding space for metric learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 471–480.

[24] K. Roth, B. Brattoli, and B. Ommer, "Mic: Mining interclass characteristics for improved metric learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 8000–8009.

[25] T. Milbich, K. Roth, H. Bharadhwaj, S. Sinha, Y. Bengio, B. Ommer, and J. P. Cohen, "Diva: Diverse visual feature aggregation for deep metric learning," in *European Conference on Computer Vision*. Springer, 2020, pp. 590–607.

[26] K. Roth, T. Milbich, and B. Ommer, "Pads: Policy-adapted sampling for visual similarity learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6568–6577.

[27] Y. Kim and W. Park, "Multi-level distance regularization for deep metric learning," in *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, 2021.

[28] H. Xuan, A. Stylianou, and R. Pless, "Improved embeddings with easy positive triplet mining," in *The IEEE Winter Conference on Applications of Computer Vision (WACV)*, March 2020.

[29] Y. Bengio, *Learning deep architectures for AI*. Now Publishers Inc, 2009.

[30] Y. Bengio, G. Mesnil, Y. Dauphin, and S. Rifai, "Better mixing via deep representations," in *International conference on machine learning*. PMLR, 2013, pp. 552–560.

[31] P. Upchurch, J. Gardner, G. Pleiss, R. Pless, N. Snavely, K. Bala, and K. Weinberger, "Deep feature interpolation for image content changes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7064–7073.

[32] Y. Wang, X. Pan, S. Song, H. Zhang, G. Huang, and C. Wu, "Implicit semantic data augmentation for deep networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019, pp. 12 635–12 644.

[33] Z. Fu, Y. Li, Z. Mao, Q. Wang, and Y. Zhang, "Deep metric learning with self-supervised ranking," in *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, 2021.

[34] K. Musgrave, S. Belongie, and S.-N. Lim, "A metric learning reality check," in *Computer Vision – ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham: Springer International Publishing, 2020, pp. 681–699.

[35] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 539–546.

[36] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[37] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese, "Deep metric learning via lifted structured feature embedding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4004–4012.

[38] S. Kim, D. Kim, M. Cho, and S. Kwak, "Proxy anchor loss for deep metric learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3238–3247.

[39] Y. Movshovitz-Attias, A. Toshev, T. K. Leung, S. Ioffe, and S. Singh, "No fuss distance metric learning using proxies," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 360–368.

[40] Q. Qian, L. Shang, B. Sun, J. Hu, H. Li, and R. Jin, "Softtriple loss: Deep metric learning without triplet sampling," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 6450–6458.

[41] X. Wang, H. Zhang, W. Huang, and M. R. Scott, "Cross-batch memory for embedding learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6388–6397.

[42] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9729–9738.

[43] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2536–2544.

[44] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *International Conference on Learning Representations*, 2018.

[45] X. Zhai, A. Oliver, A. Kolesnikov, and L. Beyer, "S4l: Self-supervised semi-supervised learning," in *Proceedings of the IEEE international conference on computer vision*, 2019, pp. 1476–1485.

[46] Y. Duan, W. Zheng, X. Lin, J. Lu, and J. Zhou, "Deep adversarial metric learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2780–2789.

[47] B. Ko and G. Gu, "Embedding expansion: Augmentation in embedding space for deep metric learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7255–7264.

[48] Y. Zhao, Z. Jin, G.-j. Qi, H. Lu, and X.-s. Hua, "An adversarial approach to hard triplet generation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 501–517.

[49] W. Zheng, Z. Chen, J. Lu, and J. Zhou, "Hardness-aware deep metric learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 72–81.

[50] G. Gu, B. Ko, and H.-G. Kim, "Proxy synthesis: Learning with synthetic classes for deep metric learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[51] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD Birds-200-2011 Dataset," California Institute of Technology, Tech. Rep. CNS-TR-2011-001, 2011.

[52] L. Zhang, Y. Zhang, X. Gu, J. Tang, and Q. Tian, "Scalable similarity search with topology preserving hashing," *IEEE Transactions on Image Processing*, vol. 23, no. 7, pp. 3025–3039, 2014.

[53] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Deep metric learning for person re-identification," in *2014 22nd International Conference on Pattern Recognition*. IEEE, 2014, pp. 34–39.

[54] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[55] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," in *Proceedings of the IEEE international conference on computer vision workshops*, 2013, pp. 554–561.

[56] Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang, "Deepfashion: Powering robust clothes recognition and retrieval with rich annotations," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1096–1104.

[57] H. Liu, Y. Tian, Y. Yang, L. Pang, and T. Huang, "Deep relative distance learning: Tell the difference between similar vehicles," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2167–2175.
[58] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.
[59] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[60] M. Rolínek, V. Musil, A. Paulus, M. Vlastelica, C. Michaelis, and G. Martius, "Optimizing rank-based metrics with blackbox differentiation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7620–7630.
[61] F. Cakir, K. He, X. Xia, B. Kulis, and S. Sclaroff, "Deep metric learning to rank," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1861–1870.
[62] L. Van Der Maaten, "Accelerating t-sne using tree-based algorithms," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3221–3245, 2014.

Zheren Fu received the B.S. degree from University of Science and Technology of China, Hefei, China, in 2020. He is currently working toward the Master degree with the University of Science and Technology of China, Hefei, China. His research interests mainly cover metric learning and image-text matching.

Zhendong Mao received the Ph.D. degree in computer application technology from the Institute of Computing Technology, Chinese Academy of Sciences, in 2014. He is currently a professor with the School of Cyberspace Science and Technology, University of Science and Technology of China, Hefei, China. He was an assistant professor with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing, from 2014 to 2018. His research interests include computer vision, natural language processing and cross-modal understanding.

Chenggang Yan received the B.S. degree in control science and engineering from Shandong University, Shandong, China, in 2008, and the Ph.D. degree in computer science from Chinese Academy of Sciences University, Beijing, China, in 2013. He is currently a Professor with the Department of Automation, Hangzhou Dianzi University. His research interests include computational photography and pattern recognition and intelligent system.

An-An Liu received the Ph.D. degree from Tianjin University in 2010. He is currently a Full Professor with the School of Electronic Engineering, Tianjin University, China. His research interests include computer vision and machine learning.

Hongtao Xie received the Ph.D. degree in computer application technology from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2012. He is currently a Research Professor with the School of Information Science and Technology, University of Science and Technology of China, Hefei, China. His research interests include multimedia content analysis and retrieval, deep learning, and computer vision.

Yongdong Zhang (Senior Member, IEEE) received the Ph.D. degree in electronic engineering from Tianjin University, Tianjin, China, in 2002. He is currently a Professor with the School of Information Science and Technology, University of Science and Technology of China, Hefei, China. He has authored more than 100 refereed journal and conference papers. His research interests include multimedia content analysis and understanding, multimedia content security, video encoding, and streaming media technology. He was the recipient of the Best Paper Awards in PCM 2013, ICIMCS 2013, and ICME 2010, and the Best Paper Candidate in ICME 2011. He serves as an Editorial Board Member for the Multimedia Systems Journal and the IEEE TRANSACTIONS ON MULTIMEDIA.