HSE: Hybrid Species Embedding for Deep Metric Learning

Bailin Yang[1,2], Haoqiang Sun[1], Frederick W. B. Li[3], Zheng Chen[1], Jianlu Cai[1], Chao Song[1*]

[1]Faculty of Computer Science and Technology, Zhejiang Gongshang University
[2]Faculty of Statistics and Mathematics, Zhejiang Gongshang University
[3]Department of Computer Science, University of Durham

Abstract

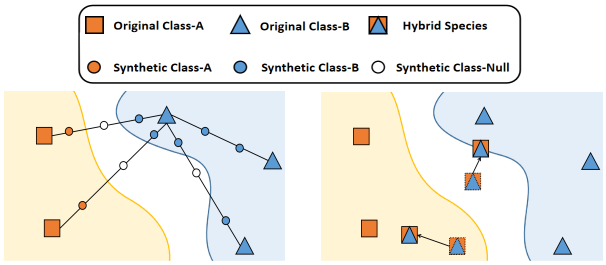
Deep metric learning is crucial for finding an embedding function that can generalize to training and testing data, including unknown test classes. However, limited training samples restrict the model's generalization to downstream tasks. While adding new training samples is a promising solution, determining their labels remains a significant challenge. Here, we introduce Hybrid Species Embedding (HSE), which employs mixed sample data augmentations to generate hybrid species and provide additional training signals. We demonstrate that HSE outperforms multiple state-of-the-art methods in improving the metric Recall@K on the CUB-200 , CAR-196 and SOP datasets, thus offering a novel solution to deep metric learning's limitations.

1. Introduction

Image retrieval heavily depends on deep metric learning to grasp visual similarities. In its early stages, methods like pair-based loss [8, 13, 19, 37] and proxy-based loss [25, 15] were proposed to train models. Nevertheless, the true goal of image retrieval is to adapt to the unknown, and the challenge lies in selecting a metric that can effectively handle differences between classes in deep metric learning. A promising approach involves exploring both intra-class and inter-class variations within the image itself, rather than relying solely on label information [33, 43, 40, 23, 51, 30].

Hard negative samples, often termed false positives, refer to images that resemble anchor images but carry different labels. The reason behind these hard samples lies in their potential for sharing remarkably similar or even identical features with the anchor samples. Approaches like hard sample mining and generation have been proposed to aid network convergence by introducing a substantial gradient [1, 41, 12, 34, 49, 48, 17, 18]. Recently, an advancement in generating appropriate hard samples involves creating supplementary training data [17, 38, 5, 21]. This is primarily achieved through linear interpolation, a prevalent method for generating synthetic samples. However, assigning absolute labels to these synthesized samples is intricate due to the manifold structures and cluster boundaries present in the embedded space [17]. Even when the new samples are formed by linear interpolation between samples of the same label, accurately determining absolute labels remains a challenge. Synthetic sample labels have emerged as a potential solution, with [38] corroborating their similarity to mixup data augmentation and highlighting the label's representation through linear interpolation. Nonetheless, this approach is constrained to mixup data augmentation, as mixup uniformly blends features and labels from different images, which might not hold true for other techniques.

We introduce a novel Hybrid Species Embedding (HSE) that leverages mixed sample data augmentations to generate additional training samples [35, 47, 45, 2, 20]. HSE creates hybrid species by combining features from multiple classes and embedding them into spatial positions. During training, different classes are randomly chosen for data augmentation



(a) Linear interpolation (b) Our HSE

Figure 1: (a) Deep metric learning methods struggle with linear interpolation for generating new training samples with labels when new samples fall between anchors of different classes. The manifold structure lacking clear boundaries makes it challenging to assign a label, even when new samples are between anchors of the same class. (b) Samples with multiple simultaneous features may not lie on a linear interpolation path, and their location may be unknown. They can be classified as an unknown class situated inside or outside a manifold boundary. To tackle this challenge, we propose using the HSE method to allocate the sample to an appropriate embedding space location based on its feature information.

*Corresponding to:csong@zjsu.edu.cn

in each batch. The labels of hybrid species are disregarded, and their placement near the category with the most similar features simulates human categorization behavior, as depicted in Figure 1(b). This strategy positions hybrid species close to the samples they are synthesized from, ensuring similarity while avoiding unrelated samples. By synthesizing samples using various mixed sample data augmentation techniques, each incorporating multiple features of a known class, our approach naturally imparts similar characteristics to the hard sample for the model without requiring explicit mining. Additionally, we dynamically adjust the position of each batch of hybrid species embedding to bypass the need for considering label information of these hybrid instances. Our contributions encompass:

• Introducing a novel metric learning strategy embedding challenging and unseen classes (Hybrid species), during training. These hybrid species offer supplementary training signals, enhancing the generalization capacity of downstream tasks, as validated through experiments.

• Adapting mixed sample data augmentation techniques for our Hybrid Species Embedding (HSE) to address constraints associated with pair-based metric learning losses, which typically demand explicit class labels.

• Demonstrating through experiments our efficacy in enhancing performance across state-of-the-art metric learning tasks on CUB-200, CAR-196, and SOP datasets.

2. Related Work

We utilize the Mixed Sample Data Augmentation (MSDA) method to create unknown samples and improve the model's generalization ability for downstream tasks. This approach resolves the issue of insignificant comparisons caused by random selection of samples in metric learning. While mining hard samples from a finite training set is limited, MSDA enables us to create more hard samples. In the following, we review existing relevant works.

2.1. Metric Learning

Deep metric learning compares data in pairs and pulls them together if they belong to the same category, otherwise separates them [8, 13, 19, 37, 26]. The exponential growth in the number of tuples during training can present challenges in achieving model convergence, and may include meaningless metrics. Proxy-based approaches, e.g., choosing a proxy point for each class, have been proposed as a solution [25, 15]. These proxies contain information from multiple samples, require less computation in pairwise comparisons, helping the network converge more effectively.

Methods that rely solely on label information may result in over-clustered or overly separated samples with identical or differing labels, respectively, and can lead to reduced performance on new and unseen test samples. EPSHN [43] notes this problem, particularly for species with varying features such as birds of the same species with different sexes. To avoid this, [30, 40, 31, 33, 51, 50, 18] emphasize the importance of incorporating semantic information from the images themselves, often through large pre-trained language models like CLIP [29] and BERT [3], to better align the visual representation space and discover potential differences within and between classes. The MSloss function [40] is one such approach that combines self-similarity and relative similarities to weight selected pairs.

Depending solely on label information can be insufficient for effective sample learning. Previous works addressed this limitation by assigning varying metrics or weights based on inter-sample variability. However, these techniques had limitations when dealing with a small number of training samples or unseen samples. To address this, we introduce additional training samples and a novel metric for improved learning. Prior works like [49, 17] added additional samples, but mainly relied on linear interpolation. In contrast, our approach generates extra training samples through selecting the appropriate MSDA, providing greater flexibility than linear interpolation.

2.2. Hard Sample Mining

Pair-based metric learning generates lots of paired samples, and handling negative samples is crucial for reducing model redundancy. Various studies [6, 10, 34, 41, 49] have explored the use of hard negative mining while constructing triples to create useful gradient sums, aiding fast convergence of triplet loss networks. Additionally, [34] suggested a semi-hard negatives scheme that selects even harder negative samples, while [6] introduced a hierarchical triplet loss (HTL) that constructs a hierarchical tree of all classes and selects semi-hard negatives with a dynamic margin.

The discussed methods require selecting hard samples from the training set to improve model performance. Hard samples are challenging examples that can evaluate the model's ability to generalize. For example, if a model performs well on horses and donkeys but struggles with mules, evaluating its performance on mules (as a hybrid of both species) can directly assess its generalization ability.

Recent studies focus on generating additional hard samples for training. HDML [49] creates hard samples between different labeled samples, while [17] generates additional training samples between identically labeled samples through hard sample mining. Gu *et al*. [7] use synthetic proxy points and competition to create hard samples. These methods use linear interpolation to generate new samples, leading to false negatives and positives due to the difficulties in determining class boundaries on the manifold structure. DAS [21] and IAA [5] try to solve this issue by adjusting features slightly and adding extra examples during mid-training of the model, aiming to make sure that new examples still belong to the same class. Despite these attempts,

completely removing this problem remains challenging.

We use mixed sample data augmentations in the preprocessing stage of images to generate additional hard samples, instead of linear interpolation. This has several advantages, including a higher probability of generating hard samples without subsequent mining and the ability to identify the underlying content of the image even without its labeling information. This makes our method highly effective for generating and processing additional hard samples.

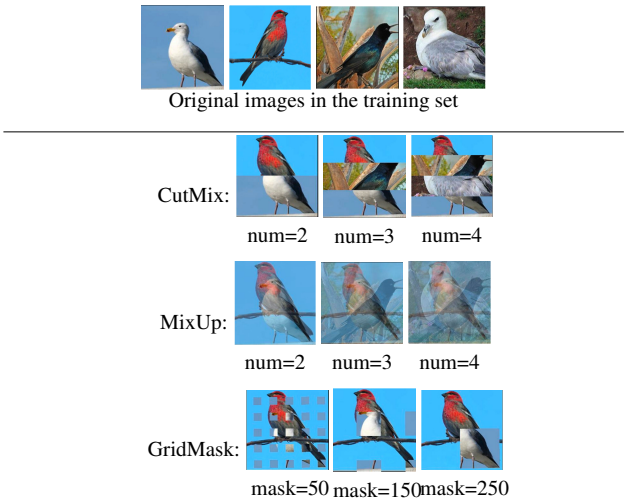


Figure 2: **Creating Hybrid Species: Three Ways.** We explore three different ways for generating hybrid species, varying parameter *num*, which controls the number of classes included. The mask indicates block size.

2.3. Mixed Sample Data Augmentations (MSDA)

Traditional data augmentation techniques assume all samples belong to the same class and do not consider relationships between different samples. In contrast, MSDA combines two samples, using various forms of label representation [35]. Popular methods include Mixup [47], Cutmix [45], Cutout [4], and GridMask [2], each with the core idea of producing images with more representative features of real-world scenarios. Other MSDA techniques include SmoothMix [20] and FMix [9].

Despite significant achievements in image classification, the above methods have not been applied to metric learning. [17] discussed the similarity between linear interpolation of features and mixup data augmentation, but due to the peculiarities of mixed labels, it cannot be directly applied to metric learning, which typically requires distinguishing samples as dichotomous, *i.e.*, positive and negative samples. Recently, [38] modified an existing metric learning loss function to accommodate mixup, but the input mix was limited to mixup. Our proposed method can be applied to a variety of mixed sample data augmentations (Section 3.2).

We modified multiple mixed sample data augmentations as shown in Figure. 2 to ensure that mixed data augmented samples contain features from multiple classes in a balanced manner. Specifically, for **CutMix**, we uniformly cropped and merged images from top to bottom to ensure that features from multiple classes are regularly distributed. For **Mixup**, we fused the samples in equal proportions to ensure a balanced representation of features from multiple classes. Lastly, we modified **Gridmask** by replacing the mask with an image to create a fusion effect.

3. Proposed Method

3.1. Preliminaries

As a preliminary for our proposed method, we consider a batch of images denoted by $x = x_1, x_2, ..., x_n$ with corresponding labels $y = y_1, y_2, ..., y_n$. Deep metric learning involves the use of a Convolutional Neural Network (CNN) to learn a non-linear transformation of each image into an m-dimensional deep feature space $\phi(x; \theta_\phi) : \mathcal{X} \rightarrow \mathbb{R}^m$, where θ_ϕ represents the network's parameters. In order to further learn a mapping from the m-dimensional feature space to a k-dimensional space, we use a linear mapping layer represented by $f(\phi; \theta_f) : \mathbb{R}^m \rightarrow \mathbb{R}^k$, where θ_f represents the parameters of the linear mapping layer.

Formally, we define the distance between two data points in the embedding space as:

$$d_f(x_i, x_j) = ||f(\phi_i) - f(\phi_j)||_2 \quad (1)$$

Metric learning models usually involve pairwise sampling of a sample, e.g. considering the contrastive loss:

$$L_{cont} = y d_f(a, x)^2 + (1 - y)[m - d_f(a, x)]_+^2 \quad (2)$$

where y denotes the label, a is the anchor image and m is the marginal value. To train a metric learning model, we first select an anchor image and choose a positive sample with the same label (y=1) and a negative sample with a different label (y=0). The network is then trained to minimize a loss function that penalizes negative samples that are too close together and positive samples that are too far apart. It is noteworthy that the loss value will be relatively large when the distance between the anchor point and the negative sample is small, while it will be relatively small when the distance is very far. Thus, selecting hard samples during the selection of pairs will result in a larger gradient.

Studies have tried to increase the number of hard samples for metric learning by generating more samples. However, linear interpolation methods can produce false positives and negatives. It is important to note that the binary metric learning label y cannot be used as a one-hot label for image classification tasks.

Our Hybrid Species Embedding (HSE) method addresses the challenges of creating additional hard samples and applying them to metric learning without label information to improve generalization. We address the challenge of

mixed sample data augmentations by emulating human categorization behavior to assign these samples to a reasonable position in the embedding space. We propose incorporating newly generated samples as an extra training signal for the model, using the same feature extraction network for both the original image and the hybrid species generated using MSDA, and included in each batch. The loss function L_m clusters the original samples based on their labels, while our proposed loss function L_{Hy} deals with the embedding positions of the new samples, as shown in Figure. 3.

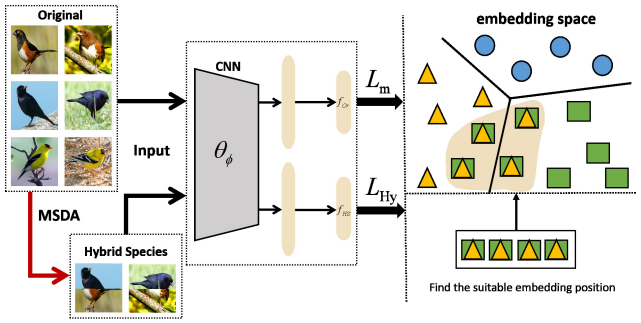
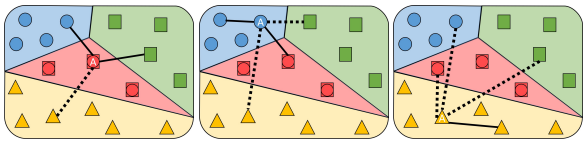


Figure 3: Our HSE method uses batches of original images and hybrid species generated by Mixed Sample Data Augmentation (MSDA). The feature extraction network (CNN) processes both types of images to obtain their corresponding feature vectors (f_{Or} and f_{HS}). The loss function includes two components: L_m (metric learning loss) groups original samples into clusters based on their labels, and L_{Hy} optimizes feature extraction by comparing hybrid species embedding positions.



(a) HS-Anchor (b) WP-Anchor (c) N-Anchor

Figure 4: **Three types of anchor point selection:** The dashed lines indicate that the sample pairs should move away from each other, while the solid lines indicate that the sample pairs should move closer to each other. The red sample is a hybrid species synthesized from green and blue. (a) denotes the use of the hybrid species as the anchor point, (b) denotes using the class that synthesizes the hybrid species as an anchor point, and (c) denotes using a class unrelated to the hybrid species as an anchor point.

3.2. Hybrid Species

In the context of image classification tasks, hard negative samples refer to images that are similar to the anchor images but have different labels. We posit that hard negative samples typically possess similar features to the anchor samples. Therefore, when a sample is derived from an anchor image, there is a high likelihood that the sample will be classified as a hard sample. Let x_A and x_B denote two images with labels y_A and y_B, respectively, where $y_A \neq y_B$. The generated sample x_H contains both the features of x_A and the features of x_B. We refer to x_H as the hybrid species.

x_H has a high likelihood of becoming a hard negative sample, while simultaneously presenting potential gradients in classes A and B. This can be achieved by utilizing mixed sample data augmentation techniques, such as mixup. Mixup generates a new training sample, denoted as (x_H, y_H), by linearly interpolating two training samples (x_A, y_A) and (x_B, y_B), where x_H and y_H are:

$$x_H = \lambda x_A + (1-\lambda)x_B \tag{3}$$

$$y_H = \lambda y_A + (1-\lambda)y_B \tag{4}$$

where λ is obtained from a beta distribution and is constrained to $[0, 1]$. To achieve a distribution of λ values that is centered around 0.5, λ should follow a beta distribution with shape parameters α and α, *e.g.*, $\lambda \sim \text{Beta}(\alpha, \alpha)$, where $\alpha \in (0, \infty)$. The generated training sample (x_H, y_H) is then used for model training with a loss function.

Many pair-based metric learning losses rely on having distinct class labels, but the output y_H is often a probability distribution per class, which is unsuitable for metric learning applications. Determining how to utilize these labels for metric learning presents a significant challenge. This difficulty is illustrated in Eq. 2.

$$L_{cont} = y d_f(a, x_H)^2 + (1-y)[m - d_f(a, x_H)]_+^2 \tag{5}$$

As the binary label $y \in \{0, 1\}$, the label for x_H is not straightforward, since it is a hybrid sample. When we consider x_H as a negative sample of the anchor point, it is not reasonable, as x_H contains the same features as the anchor samples. Keeping them away from each other could cause the model to ignore important features and focus on irrelevant ones.Therefore, the binary label y is invalid for x_H.

[38] showed that classification tasks are not fundamentally different from metric learning, and the label of the synthesized sample X_H can still be represented by linear interpolation of the labels. Eq. 5 can then be converted to:

$$\widetilde{L}_{cont} = \lambda d_f(a, x_H)^2 + (1-\lambda)[m - d_f(a, x_H)]_+^2 \tag{6}$$

Now, x_H appears with nonzero contributions in both positive and negative terms for positive-negative pairs, due to the interpolation factor $\lambda \in [0, 1]$.

However, we found that these methods are only applicable to mixup and may be subject to controversy in other forms of mixed sample data augmentations. This is due to the fact that mixup binds the images together evenly based on the assigned weight parameter, λ. The same applies to labels In contrast, cutmix synthesizes x_H by replacing a crop area of x_A with that of x_B, where the crop area of λ and $(1-\lambda)$ is taken from x_A and x_B, respectively. Therefore, the label of x_H cannot be accurately represented by

	HS-Anchor	WP-Anchor	N-Anchor
Intra-class variation	✔	✔	✖
Inter-class variation	✔	✔	✔
Conflict	✖	✔	✖

Table 1: Three types of anchor point selection are considered for inter-class and intra-class variation.

linear interpolation, as the area being replaced is indeterminate, and may contain either significant features or irrelevant content. Consequently, the contribution of the crop area to x_H remains inconclusive, making the application of linear interpolation for labeling mostly infeasible.

Instead of fully utilizing the label information of x_H, we can translate them into their appropriate spatial embedding, as we will describe in next subsection. Also, when applying MSDA to images, we need to consider how the labels are to be processed, which limits us to intermixing only two samples. In contrast, x_H does not require us to define label information, and can hence be mixed by multiple classes.

3.3. Hybrid loss

3.3.1 Anchor Point Selection and Processing

Although it is challenging to accurately assign a label to x_H obtained through MSDA, the features of x_H can be inferred based on the source images. We define $C(x_I)$ to represent the label of image x_I. When x_H is selected as an anchor, it may not be possible to find a suitable positive sample, but x_H shares semantic information with $C(x_A)$ and $C(x_B)$ to some extent. Thus, we can consider $C(x_A)$ and $C(x_B)$ as weak positive samples (WP) of x_H, with the remaining classes treated as negative samples.

During the training phase, the embedding of the spatial position of x_H plays a crucial role in the generalization of downstream tasks. Therefore, deciding how to treat x_H in pair selection becomes the first key question. Since x_H contains information for multiple classes, including it in a paired selection can provide more gradient information. We divided the pairing methods into three categories, as shown in Figure. 4: HS (hybrid species sample) as the anchor point, WP (weak positive sample) as the anchor point, and N (negative sample) as the anchor point. We compared the three methods and and summarized the findings in Table 1.

The choice of anchor point in a classification task can affect the learning of intra-class and inter-class variation. When using x_H as an anchor point, it pulls towards the WP while staying away from the rest of the class. Using WP as an anchor point keeps it close to its composite x_H to maintain identical points within the class, but may result in conflicts with the synthetic sample of x_H leading to non-converging losses. Anchoring with N treats x_H as a negative sample and moves it away from other samples, losing its purpose. Our findings recommend using HS as the anchor sample to effectively focus on both intra-class and inter-class variation while ensuring convergence.

3.3.2 Strategies for Sample Mining

In the previous subsection, we analyzed the contribution of pairwise selection to the embedding of x_H. Since x_H is randomly generated in each epoch, how to embed it more efficiently and reasonably becomes the second key issue of our study. For instance, when x_H is used as an anchor point, the corresponding WP may contain multiple classes. If x_H pulls all WP close together, clusters that are unrelated may merge, which is unreasonable. If x_H randomly selects WP that are far apart, the structure of the cluster may be destroyed. On the other hand, if negative samples that are far apart are selected, the resulting gradient will be smaller. In summary, it is vital that we choose a reasonable and valid sample for performing metric learning with HS.

Several studies have explored the advantages of hard negative mining in constructing informative gradients [12, 36, 34]. In this context, we introduce the notion of hard negatives for the hybrid species as:

$$x_{H_{hn}} = \underset{x:C(x_I)\neq C(x_A)\cap C(x_I)\neq C(x_B)}{\operatorname{argmin}} d\left(f\left(x_H\right), f\left(x_I\right)\right) \tag{7}$$

It is worth noting that $x_{H_{hn}}$ does not only yield a large gradient for x_H, but it also possesses the same potential gradient as the inputs $C(x_A)$ and $C(x_B)$ that are used to synthesize x_H.

The study by [43] shows that easy positive mining improves the generalization performance of downstream tasks. Our experiments also demonstrate the benefits of this approach for x_h. We define the easy hybrid weak positive of the hybrid species as:

$$x_{H_{ewp}} = \underset{x:C(x_I)=C(x_A)\cup C(x_I)=C(x_B)}{\operatorname{argmin}} d\left(f\left(x_H\right), f\left(x_I\right)\right) \tag{8}$$

There are two primary reasons why bringing x_H and $x_{H_{ewp}}$ closer together is beneficial. Firstly, it enables the model to concentrate on the features that are shared by both inputs, which can significantly influence the sample measurements. Secondly, this approach avoids the potential issue of pushing weak positives that are distant from each other too close together. By doing so, it preserves both the intra-class variance and the inter-class variance, which can help maintain the manifold structure of the classes in the embedding space.

Our hybrid loss function involves mapping the output of the convolutional neural network onto a unit sphere, which is a widely accepted technique. During each mini-batch, when x_H serves as the anchor, we select $x_{H_{ewp}}$ and $x_{H_{hn}}$ for comparison. As in Algorithm 1, the former corresponds

```
# Nsize: Number of samples of the same class in each batch
# n: The number of classes that synthesize x_H
# b: batch size d:dimension
# F = b×d
# Map of feature similarity between samples
Map = F × F^⊤;
# x_H as the anchor,look for weak positive sample
DWP = Map;
for each i ∈ [b − Nsize, b] do
    for each j ∈ [n ∗ Nsize, b] do
        DWP[i][j] = -1;
    end
end
# Each x_H corresponds to the value and index of the most similar WP sample
V_DWP,I_DWP = DWP.max(1) ;
# x_H as the anchor,look for negative sample
DN = Map;
for each i ∈ [b − Nsize, b] do
    for each j ∈ [0, n ∗ Nsize] do
        DN[i][j] = -1;
    end
    for each j ∈ [b − Nsize, b] do
        DN[i][j] = -1;
    end
end
# Each x_H corresponds to the value and index of the most similar negative sample
V_DN,I_DN = DN.max(1);
# Select the WP sample and negative sample parts corresponding to x_H
T = [V_DWP[-Nsize:],V_DN[-Nsize:]] ;
Hybrid loss = -α logsoftmax(T).mean();
```

Algorithm 1: Hybrid Loss

to the original sample, while the latter represents x_H. To synthesize x_H, we typically select samples from the first n classes. Then, we search for the most similar weak positive sample and negative sample for x_H, respectively.

The NCA loss function is an effective way to avoid the selection of margin hyperparameters and handle the distance between samples efficiently[46]. To ensure that hybrid species are embedded in a suitable location, we define the hybrid loss based on the NCA loss:

$$L_{Hy} = -\alpha \log \frac{1}{1 + \frac{\exp\left(f(x_H)^\top f\left(x_{H_{hn}}\right)\right)}{\exp\left(f(x_H)^\top f\left(x_{H_{ewp}}\right)\right)}} \tag{9}$$

where α is a hyper-parameter that controls the balance between bringing x_H and $x_{H_{ewp}}$ closer and pulling x_H and $x_{H_{hn}}$ farther apart. The L_{Hy} loss serves as an auxiliary loss to process additional samples and can be combined with other metric learning losses. As in Figure. 3, the final loss is the sum of the metric learning loss L_m and the L_{Hy} loss:

$$L = L_m + L_{Hy} \tag{10}$$

When incorporating our method with the contrastive loss (Eq. 2), the resulting final loss is:

$$L = L_{cont} + L_{Hy} \tag{11}$$

4. Experiments

Our HSE method uses pre-trained BN-Inception [14] and ResNet [11] models on ILSVRC 2012-CLS [32] with PyTorch. These models have learned feature representations from diverse images, reducing time and resources for new dataset training. They also perform well in image recognition tasks, making them suitable for computer vision applications. For generalization ability evaluation, we maintain consistency with base methods on image preprocessing, learning rate, and batch size. (Source code: https://github.com/SHQberserker/HSE)

Nsize examples from each class are selected per batch. We use two methods to incorporate hybrid species during training: selecting multiple classes to add hybrid species as additional examples of batch, and randomly replacing examples in the batch with hybrid species. We use the cutmix technique to create hybrid species by stitching two samples together evenly. Algorithm 1 fixes the last Nsize samples of each batch as synthetic samples.

4.1. Datasets and Metrics

We tested our framework's ability to perform well on challenging datasets: CUB-200, CARS-196, and SOP. These datasets involve fine-grained classification tasks, where there are many categories and relatively few images per category. We followed the data split approach from [22]. Specifically, for CUB-200, we used 5864 images from the first 100 categories for training and 5924 images from the last 100 categories for testing. For CARS-196, we trained on the first 98 classes and tested on the remaining images. Additionally, for SOP, we utilized the first 59,551 images (11,318 classes) for training and the remaining 60,502 images (11,316 classes) for testing.

To assess our experiments, we employed the Recall@K metrics. This measure involves selecting the K images that are closest to the query image. If there's an image with the same label as the query image, we score 1; otherwise, it's 0. Recall@K calculates the average score for all query images, with higher values indicating better retrieval performance. These metrics provide a way to quantitatively evaluate how well our model can find similar images, which is crucial for many image-based applications.

4.2. Results and Analysis

4.2.1 Comparison

Experiments compared our proposed method to existing methods, including N-pair, EPSHN, MSloss and Proxy-Anchor. Results are shown in Table 2 for CUB-200 and CARS-196 datasets, where * indicates our results under the same experimental settings. Our method improves on different approaches to varying degrees . Our method had less impact on the N-pair method as it mostly ignores intra-class variation, leading to excessive clustering between the same classes. However, we observed a significant improvement in recall@K when the EPSHN method was augmented with the additional training signal of HSE. Recall@1 improved by 2.5% and 2.3% on the challenging CUB-200 and CARS-196 datasets, respectively, where Nsize=16 and Nsize=8, respectively. Importantly, HSE-E outperformed the ensemble-based method that EPSHN had not surpassed before on the CUB-200 and CARS-196 datasets.

We believe that EPSHN tends to ignore intra-class variations, such as those found in [43] where birds of different sexes may vary greatly within the same class, but should also focus on characteristics common between members of the same class. When x_H is added, it helps the network to focus more on these common characteristics. MSloss focuses on both inter-class variation and intar-class variation. However, when adding our additional training signals, there is still a certain improvement. This is because the features that our additional training signals focus on help the network generalize to unknown classes.

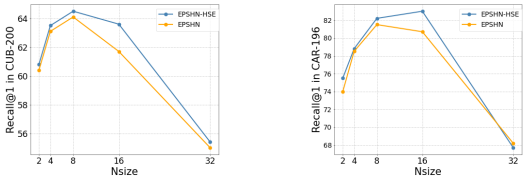
Our method achieves good results in pair-based methods and is effective in proxy-based metric learning methods. We improve approximately 1% in both Recall@1 and Recall@2. This confirms our universality, which can be applied to various metric learning loss functions. We also compared our method to the Metrix/input [38] approach. Metrix/input has a high computational cost because it calculates $\frac{1}{2}n(n-1)$ embeddings for a batch of n examples. Instead, our method achieves a similar effect using only $Nsize * Cnum$ embeddings. Our approach is hence more computationally efficient. We also remain applicable to discriminate between positives and negatives for every anchor point. We can also incorporate non-additive component functions that involve loss functions, e.g., EPSHN.

HSE is also effective on the SOP dataset. Unlike smaller datasets, the SOP dataset has superclasses, which introduce more variation between different superclasses and less variation within the same superclass. It's challenging to measure similarity between samples within the same superclass in the SOP dataset. As shown in Table 3, we generate HS from samples that share the same superclass but have different labels. When we apply HSE to various classical methods, we observe varying degrees of improvement.

4.2.2 Impact of Minibatch Parameters

Impact of Nsize

We trained a ResNet-18 embedding network on CUB-200 and CARS-196 datasets with batch size 128 and embedding dimension 512 to investigate HSE's impact on model generalization at various Nsizes. HSE's effectiveness varies with methods, datasets, and Nsizes (Figure. 5). The best performance occurs at Nsize between 4 and 16 with a batch size of 128, followed by performance degradation. When there are too few negative samples, the hybrid species' gradient information becomes insignificant, and they may be grouped with WP's clusters, harming downstream tasks.



(a) CUB (b) CAR

Figure 5: Effect of additional training signals provided by different Nsizes.

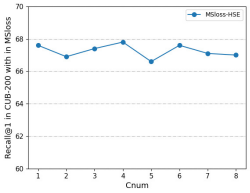


Figure 6: Results on the CUB dataset using different Cnum in MS loss.

Impact of Cnum

We found that HSE improves the generalization ability of the model. However, increasing the number of hybrid species does not necessarily lead to better results. We conducted experiments on the CUB dataset using the MS loss method, as shown in Figure. 6. Our loss function indirectly affects the entire embedding space of the batch when learning the spatial embedding of a single hybrid species. After training for a certain period, the two combined classes have essentially covered the entire dataset.

4.2.3 Effect of Hybrid Species

We investigated the effectiveness of different data augmentation techniques in providing additional training signals for hybrid species. Table 4 shows the classic blending data augmentation techniques we used, with certain modifications made to each method. Notably, CutMix and GridMask stand out. CutMix can incorporate features from each compositional image to a certain extent, but may also diminish key features. Mixup dilutes key features as the number of samples increases. GridMask (block=50) covers the entire

		CUB-200			CARS-196		
Recall@k		k=1	k=2	k=4	k=1	k=2	k=4
Angular^{512}[39]	G	53.6	65.0	75.3	71.3	80.7	87.0
HDML^{512} [49]	G	53.7	65.7	76.7	79.1	87.1	92.1
HTL^{512} [6]	BN	57.1	68.8	78.7	81.4	88.0	92.7
Margin^{128} [41]	R	63.6	74.4	83.1	79.6	86.5	91.9
SoftTriple^{512} [28]	BN	65.4	76.4	84.5	84.5	90.7	94.5
DR^{512} [24]	BN	66.1	77.0	85.1	85.0	90.5	94.1
Ensemble-based							
HDC^{384} [44]	G	53.6	65.7	77.0	73.2	82.4	86.4
A-BIER^{512} [27]	G	55.3	67.2	76.9	78.0	85.8	91.1
ABE-8^{512} [16]	G	60.6	71.5	79.8	60.6	71.5	79.8
DREML^{9216} [42]	R	63.9	75.0	83.1	86.0	91.7	95.0
Npair^{64}*(Nsize=8) [37]	R	51.3	64.2	75.3	64.2	76.0	84.6
HSE-N^{64}(Nsize=8)	R	52.5+1.2	66.0+1.8	77.4+2.1	64.9+0.7	76.6+0.6	84.7+0.1
EPSHN^{512}*(Nsize=16) [43]	R	64.9	75.3	83.5	83.1	89.7	93.6
HSE-E^{512}(Nsize=16)	R	66.9+2.0	77.4+2.1	85.5+2.0	85.4+2.3	91.2+1.5	96.9+3.3
HSE-E^{512}(Nsize=8)	R	67.4+2.5	77.7+2.4	85.7+2.2	84.8+1.7	90.6+0.9	94.3+0.7
MS† ^{512}	R	67.8	77.8	85.6	87.8	92.7	95.3
+Metrix/input[38]	R	69.0+1.2	79.1+1.3	86.0+0.4	89.0+1.2	93.4+0.7	96.0+0.7
MS^{512}*(Nsize=5) [40]	BN	65.7	77.0	86.3	81.7	88.7	93.2
HSE-M^{512}(Nsize=5)	BN	67.6+1.9	78.0+1.0	85.8-0.5	82.0+0.3	88.9+0.2	93.3+0.1
Proxy-Anchor† ^{512}	R	69.5	79.3	87.0	87.6	92.3	95.5
+Metrix/input[38]	R	70.5+1.0	81.2+1.9	87.8+0.8	88.2+0.6	93.2+0.9	96.2+0.7
Proxy-Anchor*^{512}[15]	R	69.4	79.2	87.0	88.5	92.7	95.6
HSE-PA^{512}	R	70.6+1.2	80.1+0.9	87.1+0.1	89.6+1.1	93.8+1.1	96.0+0.4

Table 2: The comparison of the Recall@K (%) performance of our proposed method HSE with several baseline methods on the CUB-200-2011 and Cars-196 datasets. The backbone networks used in the models are denoted by abbreviations: G for GoogleNet, BN for Inception with batch normalization, and R for ResNet50. The symbol † indicates the reproduced result reported by the original authors.N-pair and EPSHN use cnum=1 and batch size of 128. MSloss uses cnum=1, batch size of 80, and α=0.05. Proxy-Anchor uses cnum=1, batch size of 128, and α=2.

		SOP		
Recall@k		k=1	k=10	k=100
EPSHN^{512}*(Nsize=5) [43]	R	74.3	86.9	94.3
HSE-E^{512}(Nsize=5)	R	76.3+2.0	88.4+1.5	94.9+0.6
MS† ^{512}	R	76.9	89.8	95.9
+Metrix/input[38]	R	77.9+1.0	90.6+0.8	95.9
MS^{512}*[40] (Nsize=5)	R	78.2	90.2	96.2
HSE-M^{512}	R	78.7+0.5	90.4+0.2	96.1-0.1
Proxy-Anchor† ^{512}	R	79.1	90.8	96.2
+Metrix/input[38]	R	79.8+0.7	91.4+0.6	96.5+0.3
Proxy-Anchor*^{512}[15]	R	79.2	90.4	95.8
HSE-PA^{512}(Nsize=5)	R	80.0+0.8	91.4+1.0	96.3+0.5

Table 3: The comparison of the Recall@K (%) performance of our proposed method HSE with several baseline methods on the SOP.Where batch=120,cnum=12, and hybrid species as additional examples of batch.

	CUB		CAR	
	R@1	num	R@1	num
Cutmix	67.4	2	85.4	2
	67.2	3	85.0	3
	67.1	4	84.2	4
Mixup	67.0	2	85.3	2
	66.9	3	81.0	3
	R@1	Masksize	R@1	Masksize
Gridmask	67.4	50	84.9	50
	66.0	150	84.8	150
	67.1	250	84.0	250

Table 4: Results on the Cub dataset using different Mixed Sample Data Augmentations in EPSHN. (R@1 = Recall@1)

image despite using relatively small blocks. Results show that CutMix is the most effective approach for improving the performance of Hybrid Species (HS). We also found that adding more class features does not necessarily improve HS performance, but adding critical features that are specific to a particular class can lead to better HS results.

We investigate the selection of weak positive samples by hybrid species using the Proxy-Anchor method on the CUB dataset. Cutmix augmentation technique is used to stitch images either vertically or horizontally, with the upper part mainly containing the bird's head and the lower part containing the bird's body when stitched vertically. When stitched horizontally, the probability of the left and right sides containing the bird's head is almost the same. Our HSE approach improves model performance by focusing on essential image features. Our study on the CUB dataset demonstrates that the hybrid species prioritize the bird's body during metric learning, particularly the lower half, which is synthesized as the easy weak positive sample using

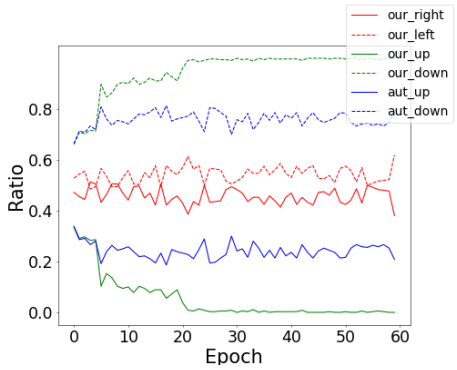


Figure 7: Analysis of easy weak positive samples selected by hybrid species using Proxy-Anchor on CUB dataset. Hybrid species are obtained by stitching two images of different labels using cutmix. We quantify the probability that selected easy weak positive samples contain upper or bottom half during training using "aut_up/down". With our method, we also report "our_up/down" and "our_left/right" indicating the likelihood of easy weak positive samples containing upper or bottom half of the bird or left or right half of the stitched image, respectively.

the Cutmix augmentation technique. While the selection probability of the lower half increases from 0.6 to 0.7∼0.8 during Proxy training, our proposed method ensures a probability of 1. Stitching hybrid species horizontally results in a selection probability of around 0.5. This approach allows the network to focus on important features during training, providing an advantage over traditional methods. Figure. 7 provides a visual representation of our findings.

4.2.4 Visualization of Embedding Space

We demonstrate the effectiveness of our proposed method on training and test data using t-SNE (t-distributed stochastic neighbor embedding) visualization technique. t-SNE can effectively reduce the dimensionality of the embedding space while preserving pairwise similarity between samples. It enables us to visually inspect how well the model separates different classes and identifies clusters of similar samples by plotting the embedding space in 2D or 3D. Therefore, t-SNE is a useful tool for evaluating the quality of learned embeddings in deep metric learning.

We experimented on two classes from the training set and two classes (wren and woodpecker) from the test set. Figure. 8 shows that EPSHN is effective at distinguishing between different classes in the training set, but the embedding for hybrid species is relatively dispersed. With the additional HSE training signal, the embedding of the hybrid species becomes more concentrated and is located between the two classes. This approach considers all the features of difficult samples, creating a more comprehensive embedding of the samples. When additional training signals are learned, the model can generalize better to unknown classes. This is demonstrated in the test set, where the EPSHN method cannot distinguish the black classes from the green and blue classes. However, when the training signal of HSE is applied to the downstream task, black classes can be successfully distinguished from green classes.

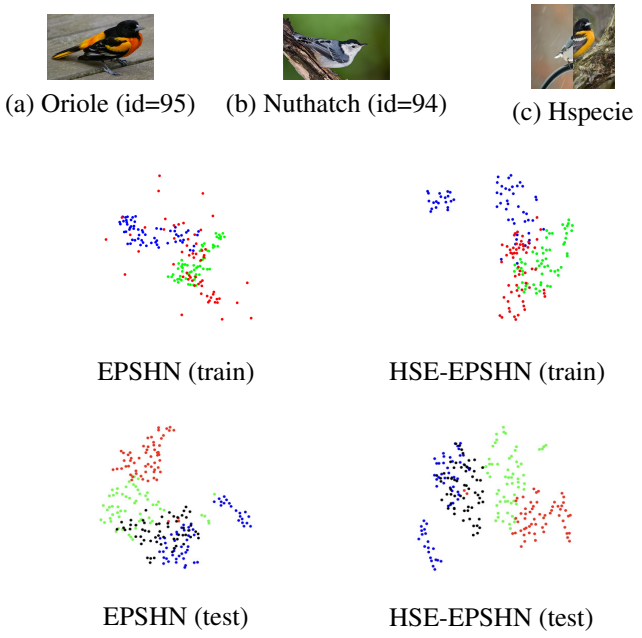


Figure 8: We present a t-SNE visualization of EPSHN and HSE-EPSHN on the test set of CUB. Specifically, we select two different classes (denoted by blue dots and green dots) from the training set as shown in (a) and (b), respectively. (c) is the synthetic class from classes (a) and (b), denoted by red dots.For the test set, we selected four bird categories, consisting of two types of wrens and two types of woodpeckers, where each color represents a specific category.

5. Conclusion

Our Hybrid Species Embedding (HSE), a metric learning approach that aims to enable models to learn the spatial embedding of unknown and difficult classes during training. HSE generates additional hard samples using mixed data augmentation methods without defining their label information and applies them to the metric learning process. Experiments demonstrate significant improvements on multiple datasets when compared to three widely used pair-based methods and state-of-the-art proxy methods. HSE enhances the generalization ability of the model by providing additional training signals in deep metric learning.

Acknowledgment. This work is partly supported by the National Natural Science Foundation of China (No: 62172366) and the Pioneer and Leading Goose R&D Program of Zhejiang Province (2023C01150).

References

[1] E. Ahmed, M. Jones, and T. K. Marks. An improved deep learning architecture for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3908–3916, 2015.

[2] P. Chen, S. Liu, H. Zhao, and J. Jia. Gridmask data augmentation. *arXiv preprint arXiv:2001.04086*, 2020.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[4] T. DeVries and G. W. Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[5] Z. Fu, Z. Mao, B. Hu, A.-A. Liu, and Y. Zhang. Intra-class adaptive augmentation with neighbor correction for deep metric learning. *IEEE Transactions on Multimedia*, 2022.

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

[7] 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*, volume 35, pages 1460–1468, 2021.

[8] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[9] E. Harris, A. Marcu, M. Painter, M. Niranjan, A. Prügel-Bennett, and J. Hare. Fmix: Enhancing mixed sample data augmentation. *arXiv preprint arXiv:2002.12047*, 2020.

[10] B. Harwood, V. Kumar BG, G. Carneiro, I. Reid, and T. Drummond. Smart mining for deep metric learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2821–2829, 2017.

[11] 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*, pages 770–778, 2016.

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

[13] E. Hoffer and N. Ailon. Deep metric learning using triplet network. In *International workshop on similarity-based pattern recognition*, pages 84–92. Springer, 2015.

[14] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[15] S. Kim, D. Kim, M. Cho, and S. Kwak. Proxy anchor loss for deep metric learning. *IEEE*, 2020.

[16] 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)*, pages 736–751, 2018.

[17] 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*, pages 7255–7264, 2020.

[18] B. Ko, G. Gu, and H.-G. Kim. Learning with memory-based virtual classes for deep metric learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11792–11801, 2021.

[19] M. T. Law, N. Thome, and M. Cord. Quadruplet-wise image similarity learning. In *Proceedings of the IEEE international conference on computer vision*, pages 249–256, 2013.

[20] J.-H. Lee, M. Z. Zaheer, M. Astrid, and S.-I. Lee. Smoothmix: a simple yet effective data augmentation to train robust classifiers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 756–757, 2020.

[21] L. Liu, S. Huang, Z. Zhuang, R. Yang, M. Tan, and Y. Wang. Das: Densely-anchored sampling for deep metric learning. In *European Conference on Computer Vision*, pages 399–417. Springer, 2022.

[22] 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*, pages 1096–1104, 2016.

[23] 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*, pages 590–607. Springer, 2020.

[24] D. D. Mohan, N. Sankaran, D. Fedorishin, S. Setlur, and V. Govindaraju. Moving in the right direction: A regularization for deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14591–14599, 2020.

[25] Y. Movshovitz-Attias, A. Toshev, T. K. Leung, S. Ioffe, and S. Singh. No fuss distance metric learning using proxies. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[26] 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*, pages 4004–4012, 2016.

[27] M. Opitz, G. Waltner, H. Possegger, and H. Bischof. Deep metric learning with bier: Boosting independent embeddings robustly. *IEEE transactions on pattern analysis and machine intelligence*, 42(2):276–290, 2018.

[28] 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/CVF International Conference on Computer Vision*, pages 6450–6458, 2019.

[29] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[30] K. Roth, O. Vinyals, and Z. Akata. Integrating language guidance into vision-based deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16177–16189, 2022.

[31] K. Roth, O. Vinyals, and Z. Akata. Non-isotropy regularization for proxy-based deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7420–7430, 2022.

[32] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[33] A. Sanakoyeu, V. Tschernezki, U. Buchler, and B. Ommer. Divide and conquer the embedding space for metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 471–480, 2019.

[34] 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*, pages 815–823, 2015.

[35] C. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[36] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *Proceedings of the IEEE international conference on computer vision*, pages 118–126, 2015.

[37] K. Sohn. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*, 29, 2016.

[38] S. Venkataramanan, B. Psomas, Y. Avrithis, E. Kijak, L. Amsaleg, and K. Karantzalos. It takes two to tango: Mixup for deep metric learning. *arXiv preprint arXiv:2106.04990*, 2021.

[39] J. Wang, F. Zhou, S. Wen, X. Liu, and Y. Lin. Deep metric learning with angular loss. In *Proceedings of the IEEE international conference on computer vision*, pages 2593–2601, 2017.

[40] 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/CVF Conference on Computer Vision and Pattern Recognition*, pages 5022–5030, 2019.

[41] 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*, pages 2840–2848, 2017.

[42] H. Xuan, R. Souvenir, and R. Pless. Deep randomized ensembles for metric learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 723–734, 2018.

[43] H. Xuan, A. Stylianou, and R. Pless. Improved embeddings with easy positive triplet mining. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2474–2482, 2020.

[44] Y. Yuan, K. Yang, and C. Zhang. Hard-aware deeply cascaded embedding. In *Proceedings of the IEEE international conference on computer vision*, pages 814–823, 2017.

[45] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019.

[46] D. Zhang, Y. Li, and Z. Zhang. Deep metric learning with spherical embedding. *Advances in Neural Information Processing Systems*, 33:18772–18783, 2020.

[47] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[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)*, pages 501–517, 2018.

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

[50] W. Zheng, C. Wang, J. Lu, and J. Zhou. Deep compositional metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9320–9329, 2021.

[51] W. Zheng, B. Zhang, J. Lu, and J. Zhou. Deep relational metric learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12065–12074, 2021.