

# Deep Variational Metric Learning

Xudong Lin, Yueqi Duan, Qiyuan Dong, Jiwen Lu\*, and Jie Zhou

Tsinghua University, Beijing 100084, China  
stein6@163.com, {duanyq14, dqy15}@mails.tsinghua.edu.cn, {lujiwen,  
jzhou}@tsinghua.edu.cn

**Abstract.** Deep metric learning has been extensively explored recently, which trains a deep neural network to produce discriminative embedding features. Most existing methods usually enforce the model to be indiscriminating to intra-class variance, which makes the model over-fitting to the training set to minimize loss functions on these specific changes and leads to low generalization power on unseen classes. However, these methods ignore a fact that in the central latent space, the distribution of variance within classes is actually independent on classes. In this paper, we propose a deep variational metric learning (DVML) framework to explicitly model the intra-class variance and disentangle the intra-class invariance, namely, the class centers. With the learned distribution of intra-class variance, we can simultaneously generate discriminative samples to improve robustness. Our method is applicable to most of existing metric learning algorithms, and extensive experiments on three benchmark datasets including CUB-200-2011, Cars196 and Stanford Online Products show that our DVML significantly boosts the performance of currently popular deep metric learning methods.

**Keywords:** Metric learning · Variational auto-encoder · Discriminative samples generating.

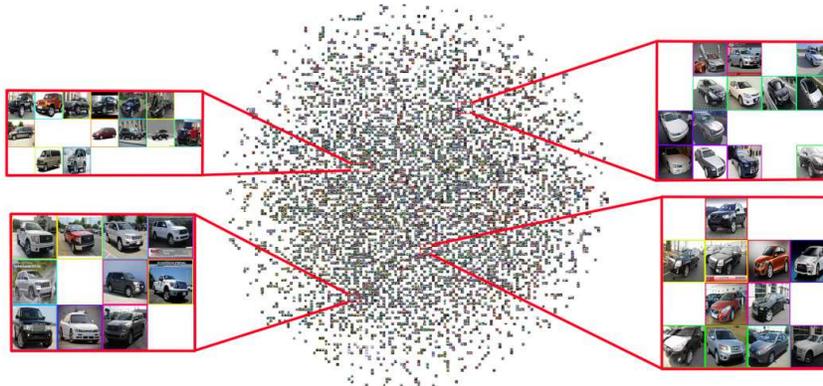
## 1 Introduction

Metric learning aims to learn a mapping with covariant relationship of distance. A good metric produces embeddings where samples from the same class have small distances and samples from different classes have large distances. Recent supervised metric learning methods uncover the potential of deep convolutional neural networks as the nonlinear mapping function through designing sampling algorithms [10,34,18,21,24,8,40,37] or modifying loss functions [36,18,7,23,33,37,3]. These methods usually share the same motivation of better maximizing inter-class distance and minimize intra-class distance. Behind this motivation, there is actually a basic assumption that every sample from the same class shares the same embedding feature. However, is this assumption really accurate?

In this paper, we provide a negative answer. Indeed, there are intra-class variances, such as pose, view point, illumination, etc, and a robust model should

---

\* Corresponding author



**Fig. 1.** Our insight: in the central latent space, the distribution of intra-class variance is independent on classes. This is the visualization of central latent space of features learned with the N-pair loss [23] using Barnes-Hut t-SNE [30] on the Cars196 test set. The color of the bounding box for each image represents the class label. Here we construct central latent space through subtracting samples’ class centers from their features. We assumed and verified that similar change of original images, like the same pose change or the same view-point change, affects their features in a similar way. (Best viewed when zoomed in.)

be able to handle these variances. However given a limited training set, deep models will easily be over-fitting if we force it to be indiscriminating to these intra-class variances. For example, in image classification, if the illumination of the object region varies too much among different samples, the model would probably be trained to ignore these important parts but to classify the training samples from their backgrounds. This leads to poor generalization ability. Therefore this assumption is ideal but not practical.

Our insight is that the distribution of intra-class variance is actually independent on classes. It is obvious that for each class, the possible intra-class variances are from exactly the same set. As presented in Fig. 1, a similar pose change in samples from different classes leads to a cluster in the central latent space. Furthermore, if we know the distribution of intra-class variance, then we can generate potential hard samples from easy samples by adding intra-class variance to it. Therefore we can confidently propose our modified assumptions: Embedding features of samples from the same class consist of two parts; One represents the intra-class invariance, and the other represents the intra-class variance which obeys the identical distribution among different classes.

In this paper, we propose a deep variational metric learning (DVML) framework following this assumption. Utilizing variational inference, we can force the conditional distribution of intra-class variance, given a certain image sample, to be isotropic multivariate Gaussian. Moreover, we can utilize most of the current metric learning algorithms to train the intra-class invariance. To be specific, the training procedure of DVML is simultaneously constrained by the following four

loss functions: 1) the KL divergence between learned distribution and isotropic multivariate Gaussian; 2) the reconstruction loss of original images and images generated by the decoder; 3) the metric learning loss of learned intra-class invariance; 4) the metric learning loss of the combination of sampled intra-class variance and learned intra-class invariance. The first two losses ensure that the intra-class variance shares the same distribution and does not contain sample-specific information of each image sample. The third ensures that the intra-class invariance represents a good class center for each class, and the fourth ensures a robust boundary among classes.

To the best of our knowledge, this is the first work that utilizes variational inference to disentangle intra-class variance and leverages the distribution to generate discriminative samples to improve robustness. It is noticeable that our framework is also applicable to hard negative mining methods. Additionally, experimental results on three benchmark datasets including CUB-200-2011, Cars196 and Stanford Online Products, show that DVML<sup>1</sup> significantly boosts the performance of existing deep metric learning algorithms.

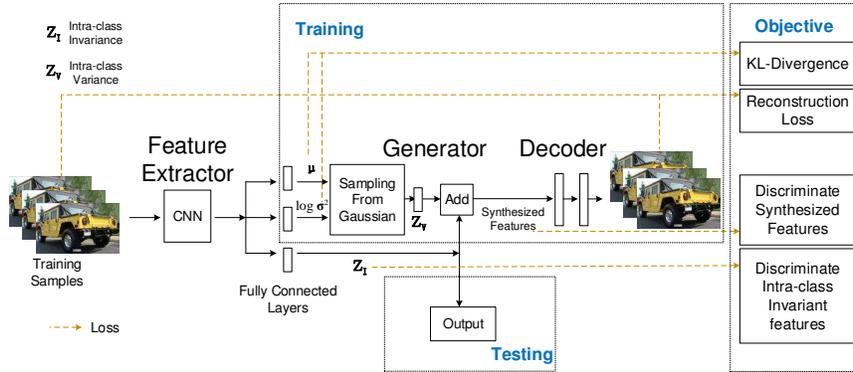
## 2 Related Work

Great progress has been made about metric learning [6,10,35,9,41,39,15] recently. In the conventional metric learning algorithms, our goal is to learn a linear Mahalanobis distance to measure the similarities of samples [19,20,5,36,1]. Some of the previous works [38,32] also tried to formulate metric learning as a variational inference problem, while focusing on the distribution of pairwise distance. There are also attempts on combining latent variables and metric learning [26], while in this work latent variables are the features of patches cropped from images.

Recently, metric learning with deep neural networks has been densely explored. There are mainly two subjects: sampling methods and loss functions. By sampling methods [10,34,18,21,24,8,40,37], we aim to mine samples which improve robustness. For example, Wu *et al.* [37] proposed a distance weighted sampling method. By loss functions [36,18,7,23,33,37,3], we aim to fully use the data in a mini-batch to learn a discriminative boundary among classes. For example, Song *et al.* [23] presented a N-pair loss which takes advantage of the whole training batch. There are also works about synthesized negative samples. In [16], they generated a proxy for each class which represents the tight upper bound of the class. However, this is different from our generating hard samples simultaneously in the training procedure, which better uncovers the potential of easy negative samples. Inspired by the central limit theorem and recent works [12,14,29,2], we begin to think about the invariance among classes.

In [4], they also model intra-class variance with an isotropic Gaussian, while it is based on the assumption that each class shares the same prior probability and covariance, which is aimed to tackle the imbalance among samples in long-tailed datasets. The core distinction is that we disentangle intra-class variance

<sup>1</sup> Code is coming soon on <https://github.com/XudongLinthu>



**Fig. 2.** Our proposed DVML framework. Taking the output of a backbone feature extractor as input, the following layers consist of two parts. The upper part is to model intra-class variance, and it only works in the training procedure. The third fully connected layers following the feature extractor is used to learn intra-class invariant features  $z_I$ , namely, the class centers, which is also the output of our model. The generator takes as inputs the class centers  $z_I$  and the features sampled from the learned distribution  $\mathcal{N}(z_V; \mu^{(i)}, \sigma^{2(i)}\mathbf{I})$ , and then outputs element-wise sum of them as synthesized discriminative samples. In order to reduce computation cost, we reconstruct the 1024-dimension features, which are the output of the backbone feature extractor, instead of the whole images, . (Best viewed when zoomed in.)

and class centers. In [4], they only learn the conditional probability of belonging to a certain class given the input image. Instead, DVML is the combination of a discriminative model and a generative model, where the former outputs class centers and the latter fits intra-class variance. Our DVML is able to boost current metric learning methods by disentangling intra-class variance and class centers, and generating potentially hard and positive samples.

### 3 Proposed Approach

In the conventional metric learning methods, intra-class variance and class centers are entangled, which brings two limitations to further improvement of metric learning algorithms:

- Given a limited dataset with a large range of variance within classes, current metric learning methods are easily over-fitting and lose discriminative power on unseen classes;
- Without disentangling intra-class variance and class centers, current methods learn a metric by exploring the boundary among classes, which means numerous easy negative samples contribute little to the training procedure.

We explored a way beyond these two limitations, with the proposed deep variational metric learning (DVML) framework. In this section, we first review

current deep metric learning methods, and introduce the variational inference for intra-class variance distribution. After explaining discriminative sample generation, we give the whole picture of deep variational metric learning. In the end, we introduce the implementation details.

### 3.1 Preliminaries

Most of recently popular deep metric learning algorithms optimize an appropriate objective function  $L$  to get the parameters of a deep neural network  $F$ .

$$\phi_F = \arg \min_{\phi_F} L_m(\phi_F; \mathbf{X}, F), \quad (1)$$

Here  $\mathbf{X}$  represents the whole training set. In the training procedure, we usually construct mini-batches of training data,  $X_b$ . Based on different ways of constructing mini-batches, various types of objective functions are designed. There are mainly three types of methods to construct mini-batches: pair-based, triplet-based, and batch-based.

In pair-based mini-batch construction, a mini-batch consists of pairs of positive and negative samples,  $\mathbf{x}_p$  and  $\mathbf{x}_n$ . In triplet-based mini-batch construction, a mini-batch consists of triplets. In a triplet, there are three samples, the negative  $\mathbf{x}_n$ , the positive  $\mathbf{x}_p$ , and the anchor  $\mathbf{x}_a$ . The positive and the anchor have the same class label, and the negative is from other classes. In batch-based mini-batch construction, we know each sample's class information. Many hard negative mining algorithms are also batch-based, for they usually have to leverage class information to mine hard pairs or triplets within a mini-batch.

With these mini-batch construction methods, most of current objective functions aim to enforce the negative samples to be away from positive ones. We utilized the following loss as our baseline methods.

Triplet-based, Triplet [36,18]:

$$L_m = \sum_{i=1}^N \max(\alpha + D(\mathbf{z}_{(a)}^{(i)}, \mathbf{z}_{(p)}^{(i)})^2 - D(\mathbf{z}_{(a)}^{(i)}, \mathbf{z}_{(n)}^{(i)})^2, 0), \quad (2)$$

where  $\mathbf{z}_{(p)}^{(i)} = F(\mathbf{x}_{(p)}^{(i)})$ ,  $\mathbf{z}_{(a)}^{(i)} = F(\mathbf{x}_{(a)}^{(i)})$ , and  $\mathbf{z}_{(n)}^{(i)} = F(\mathbf{x}_{(n)}^{(i)})$ . Here  $\mathbf{x}_{(p)}^{(i)}$ ,  $\mathbf{x}_{(n)}^{(i)}$ ,  $\mathbf{x}_{(a)}^{(i)}$  denote the positive, the negative, and the anchor samples.  $N$  is the number of triplets.  $D(\mathbf{z}_{(p)}^{(i)}, \mathbf{z}_{(n)}^{(i)})$  is the distance between features embedded from image samples.

Batch-based, N-pair [23]:

$$L_m = \frac{1}{N} \sum_{i=1}^N \log(1 + \sum_{j \neq i} \exp(\mathbf{z}^{(i)T} \mathbf{z}_+^{(j)} - \mathbf{z}^{(i)T} \mathbf{z}_+^{(i)})), \quad (3)$$

where  $\mathbf{z}_i = F(\mathbf{x}^{(i)})$ , and the batch consists of  $\mathbf{x}$  and  $\mathbf{x}_+$ . Here  $\mathbf{x}^{(i)}$  and  $\mathbf{x}_+^{(j)}$  are from the same class, only when  $i = j$ .

Batch-based, Triplet<sub>2</sub> with Distance Weighted Sampling [37]:

$$L_m = \sum_{i=1}^N \max(\alpha + D(\mathbf{z}_{(a)}^{(i)}, \mathbf{z}_{(p)}^{(i)}) - D(\mathbf{z}_{(a)}^{(i)}, \mathbf{z}_{(n)}^{(i)}), 0), \quad (4)$$

where  $\mathbf{z}_{(a)}^{(i)}$ 's,  $\mathbf{z}_{(p)}^{(i)}$ 's and  $\mathbf{z}_{(n)}^{(i)}$ 's original image samples are determined by distance weighted sampling over a mini-batch. According to [37], this loss performed better than aforementioned triplet loss.

### 3.2 Variational Inference for Intra-Class Variance

With our insight that the distribution of intra-class variance is inter-class invariant, we can disentangle intra-class variance and intra-class invariance. Therefore our model can more explicitly learn appropriate class centers, and has the nature of robustness toward a large range of intra-class variance.

As it is hard to directly represent intra-class variance without extra annotations about pose, view point, illumination, etc, we refer to the setting of generative models. It is natural to believe that from the sum of good intra-class variance and class centers, we can reconstruct the original image.

To be specific, given  $\mathbf{X} = (\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n)})$  as the dataset, consisting of  $N$  i.i.d images from  $M$  classes, we assume the data are generated by a random process, involving an unobserved continuous random variable  $\mathbf{z} = \mathbf{z}_V + \mathbf{z}_{I_k}$ , which is actually the embedding features of given samples. The process consists of three steps: (1) a value  $\mathbf{z}_V^{(i)}$  is generated from some conditional distribution  $\mathbf{p}_\theta^*(\mathbf{z})$ , which is intra-class variance of sample  $i$  from class  $k$ ; (2)  $\mathbf{z}_{I_k}^{(i)}$  is the intra-class invariance of sample  $i$  from class  $k$ , and  $\mathbf{z}^{(i)}$  equals the sum of  $\mathbf{z}_V^{(i)}$  and  $\mathbf{z}_{I_k}^{(i)}$ ; (3) an image  $\mathbf{x}^{(i)}$  is generated from some conditional distribution  $\mathbf{p}_\theta^*(\mathbf{x}|\mathbf{z})$ .

Here, we assume that the prior  $\mathbf{p}_\theta^*(\mathbf{z})$  and the the likelihood  $\mathbf{p}_\theta^*(\mathbf{x}|\mathbf{z})$  is generated from some parametric families of distributions  $\mathbf{p}_\theta(\mathbf{z})$  and  $\mathbf{p}_\theta(\mathbf{x}|\mathbf{z})$ . As we simultaneously learn the intra-class variance and intra-class invariance, so here for sample  $i$  from class  $k$ , its  $\mathbf{z}_{I_k}^{(i)}$  is deterministic. Therefore all the distributions related  $\mathbf{z}$  could be taken as the distribution related to  $\mathbf{z}_V$ . Using Monte Carlo estimator similar to VAE [12], we can get the approximated loss for the modeling of intra-class variance,

$$L(\theta, \phi; \mathbf{x}^{(i)}) \approx -D_{KL}(q_\phi(\mathbf{z}_V|\mathbf{x}^{(i)})||p_\theta(\mathbf{z}_V)) + \frac{1}{L} \sum_{l=1}^L \log p_\theta(\mathbf{x}^{(i)}|\mathbf{z}_V^{(i,l)}). \quad (5)$$

Here we let the prior distribution of  $\mathbf{z}_V$  is the centered isotropic multivariate Gaussian  $\mathbf{p}_\theta(\mathbf{z}_V) = \mathcal{N}(\mathbf{z}_V; \mathbf{0}, \mathbf{I})$ . For the approximation posterior, we let it be a multivariate Gaussian with a diagonal covariance.

$$\log q_\phi(\mathbf{z}_V|\mathbf{x}^{(i)}) = \mathcal{N}(\mathbf{z}_V; \boldsymbol{\mu}^{(i)}, \boldsymbol{\sigma}^{2(1)}\mathbf{I}). \quad (6)$$

We use the outputs of fully-connected layers, to approximate the mean and s.d. of the posterior,  $\boldsymbol{\mu}^{(i)}$  and  $\boldsymbol{\sigma}^{(i)}$ . With the reparameterization trick, we can finally get the first two terms of our objective.

$$\begin{aligned} \mathcal{L}(\theta, \phi; \mathbf{X}_b) &\approx \frac{1}{2B} \sum_{i=1}^B \sum_{j=1}^J (1 + \log((\sigma_j^{(i)})^2) - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2) \\ &\quad + \frac{1}{TB} \sum_{i=1}^B \sum_{t=1}^L \log p_\theta(\mathbf{x}^{(i)} | \mathbf{z}^{(i,t)}) \\ &\triangleq L_1 + L_2, \end{aligned} \quad (7)$$

where  $T$  is the number of generating iterations and  $B$  is the batch-size of the mini-batch.  $L_1$  enforces the distribution of intra-class variance to be isotropic centered Gaussian, and  $L_2$  ensures the intra-class variance preserve sample-specific information. Derivation details are in the supplementary materials.

Furthermore, for simplicity, in the training procedure, we utilize L-2 distance instead of original maximum likelihood estimation to handle the decoding term  $p_\theta(\mathbf{x}^{(i)} | \mathbf{z}^{(i,t)})$ , which gives us a simplified term:

$$L_1 = \frac{1}{2B} \sum_{i=1}^B \sum_{j=1}^J (1 + \log((\sigma_j^{(i)})^2) - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2), \quad (8)$$

$$L_2 = \frac{1}{TB} \sum_{i=1}^B \sum_{t=1}^T \|\mathbf{x}^{(i)} - \hat{\mathbf{x}}^{(i,t)}\|_2. \quad (9)$$

$\mathbf{x}^{(i)}$  represents original image samples, and  $\hat{\mathbf{x}}^{(i,t)}$  is the fake sample synthesized from the sum of intra-class invariance features and intra-class variance features sampled from the distribution  $\mathcal{N}(\mathbf{z}_V; \boldsymbol{\mu}^{(i)}, \boldsymbol{\sigma}^{2(i)}\mathbf{I})$ .

### 3.3 Discriminative Sample Generation

As we addressed previously, most of current metric learning algorithms cannot uncover the full potential of easy samples. However, with the learned distribution of intra-class variance, we can generate potential hard samples from easy negative samples by adding the embedding features of easy samples with an biased term sampled from the distribution of intra-class variance.

Since we have learned an approximated conditional distribution of the intra-class variance,  $\mathcal{N}(\mathbf{z}_V; \boldsymbol{\mu}^{(i)}, \boldsymbol{\sigma}^{2(i)}\mathbf{I})$ , an idea is naturally raised: we can also draw samples from this distribution to construct synthesized embedding features, and take them as the inputs of metric learning loss functions.

$$L_3 = L_m(\hat{\mathbf{z}}), \quad (10)$$

where  $\hat{\mathbf{z}} = \mathbf{z}_{I_k} + \hat{\mathbf{z}}_V$ , and  $\hat{\mathbf{z}}_V$  is sampled intra-class variance features. By  $\mathbf{z}_{I_k}$ , we want to stress that different classes have different intra-class variance, yet we do not compute class centers over classes.

**Table 1.** Comparisons of clustering and retrieval performance (%) on the Cars196 dataset

| Method                              | NMI         | F <sub>1</sub> | R@1         | R@2         | R@4         | R@8         |
|-------------------------------------|-------------|----------------|-------------|-------------|-------------|-------------|
| Triplet [36,18]                     | 56.2        | 21.3           | 58.5        | 68.8        | 77.1        | 84.2        |
| <b>DVML+Triplet</b>                 | <b>61.1</b> | <b>28.2</b>    | <b>64.3</b> | <b>73.7</b> | <b>79.2</b> | <b>85.1</b> |
| N-pair [23]                         | 62.9        | 31.9           | 72.3        | 79.9        | 86.8        | 90.9        |
| <b>DVML+N-pair</b>                  | <b>66.0</b> | <b>34.6</b>    | <b>80.4</b> | <b>85.8</b> | <b>91.8</b> | <b>95.1</b> |
| Contrastive [7]                     | 44.8        | 11.2           | 35.8        | 47.5        | 59.7        | 71.5        |
| Lifted [25]                         | 60.0        | 27.9           | 70.0        | 79.5        | 86.8        | 92.0        |
| Angular [33]                        | 61.2        | 30.8           | 70.1        | 80.2        | 86.7        | 91.6        |
| Triplet <sub>2</sub> +DWS [37]      | 65.4        | 34.3           | 78.9        | 85.6        | 91.0        | 94.7        |
| <b>DVML+Triplet<sub>2</sub>+DWS</b> | <b>67.6</b> | <b>36.8</b>    | <b>82.0</b> | <b>88.4</b> | <b>93.3</b> | <b>96.3</b> |
| HDC [40]                            | -           | -              | 73.7        | 83.2        | 89.5        | 93.8        |
| Proxy-NCA [16]                      | 64.9        | -              | 73.2        | 82.4        | 86.4        | 88.7        |

Remembering that the distribution of intra-class variance is independent on classes, we confidently conclude that these synthesized embedding features contain a larger range of intra-class variance than original samples, and training with them will bring us a more robust model. Here is a simple example. In class  $A$ , the original samples only contain view point changes, and in class  $B$ , the original samples only contain illumination changes. Different from current models trained with only their original samples, our model is also robust to an unseen class which contains both view point changes and illumination changes.

### 3.4 Deep Variational Metric Learning

Finally we have the whole picture of our proposed DVML framework. Besides aforementioned three terms of loss functions, our final objective also contains a constraint term of intra-class invariance:

$$L_4 = L_m(\mathbf{z}_I), \quad (11)$$

where  $\mathbf{z}_I$  is the intra-class invariance features and also the output of our model in testing. This term enforces the intra-class invariance, namely the class centers, to be discriminative. It is noticeable that here we do not calculate class centers. We call them class centers for we have disentangled this part from the intra-class variance. Therefore our method is applicable to most of current deep metric learning algorithms.

The final objective function is:

$$L = \lambda_1 L_1 + \lambda_2 L_2 + \lambda_3 L_3 + \lambda_4 L_4. \quad (12)$$

**Table 2.** Comparisons of clustering and retrieval performance (%) on the Stanford Online Products dataset

| Mehtod                              | NMI         | F <sub>1</sub> | R@1         | R@10        | R@100       |
|-------------------------------------|-------------|----------------|-------------|-------------|-------------|
| Triplet [36,18]                     | 86.5        | 20.2           | 54.9        | 71.5        | 85.2        |
| <b>DVML+Triplet</b>                 | <b>89.0</b> | <b>31.1</b>    | <b>66.5</b> | <b>82.3</b> | <b>91.8</b> |
| N-pair [23]                         | 87.9        | 27.1           | 66.4        | 82.9        | 92.1        |
| <b>DVML+N-pair</b>                  | <b>90.2</b> | <b>37.1</b>    | <b>70.0</b> | <b>85.1</b> | <b>93.7</b> |
| Contrastive [7]                     | 83.5        | 10.4           | 37.4        | 52.7        | 69.4        |
| Lifted [25]                         | 88.4        | 30.6           | 65.2        | 81.3        | 91.7        |
| Angular [33]                        | 87.7        | 26.4           | 66.8        | 82.8        | 92.0        |
| Triplet <sub>2</sub> +DWS [37]      | 89.0        | 31.1           | 66.8        | 82.0        | 91.0        |
| <b>DVML+Triplet<sub>2</sub>+DWS</b> | <b>90.8</b> | <b>37.2</b>    | <b>70.2</b> | <b>85.2</b> | <b>93.8</b> |
| HDC [40]                            | -           | -              | 69.5        | 84.4        | 92.8        |
| Proxy-NCA [16]                      | -           | -              | 73.7        | -           | -           |

By simply applying sampling methods to both original features and synthesized features, or replace  $L_m$  with custom loss functions, we can combine our method with most of current metric learning approaches.

Here we want to highlight our contributions. First, to the best of our knowledge, this is the first work to disentangle intra-class variance and intra-class invariance, which makes it possible to explicitly learn appropriate class centers by simultaneously minimizing  $L_4$ . Second, different from previous hard negative mining methods which ignore numerous easy negative samples, with the learned distribution of intra-class variance, we generate discriminative samples which contains the possible intra-class variance over the whole training set. It is obvious that our discriminative sample generation is entirely different from conventional data augmentation methods. We simultaneously generate latent variables, namely, the embedding features, in the training procedure. More importantly, our synthesized samples have the variance of the whole training set.

### 3.5 Implementation Details

We implement all the compared baseline methods and our methods on Chainer [28], with the GoogLeNet [27] pre-trained on ILSVRC2012 [17] as the backbone for a fair comparison. Following standard pre-processing of data, we first normalize the images into  $256 \times 256$ , and then we perform random crop and horizontal mirroring for data augmentation. We add three parallel fully-connected layers after the average pooling layer of GoogLeNet, with same output dimension which is the required embedding size. Two of them are used to approximate  $\mu$  and  $\log \sigma^2$ . The other’s output is the intra-class variance. For the reconstruction part, due to the high cost of image reconstruction, we use the output features of

**Table 3.** Comparisons of clustering and retrieval performance (%) on the CUB-200-2011 dataset

| Mehtod                              | NMI         | F <sub>1</sub> | R@1         | R@2         | R@4         | R@8         |
|-------------------------------------|-------------|----------------|-------------|-------------|-------------|-------------|
| Triplet [36,18]                     | 52.7        | 19.8           | 39.8        | 51.9        | 63.7        | 74.5        |
| <b>DVML+Triplet</b>                 | <b>55.5</b> | <b>25.0</b>    | <b>43.7</b> | <b>56.0</b> | <b>67.8</b> | <b>76.9</b> |
| N-pair [23]                         | 60.1        | 27.1           | 50.6        | 63.1        | 73.8        | 82.2        |
| <b>DVML+N-pair</b>                  | <b>61.1</b> | <b>28.5</b>    | <b>52.5</b> | <b>64.9</b> | <b>75.3</b> | <b>84.1</b> |
| Contrastive [7]                     | 43.8        | 11.0           | 32.8        | 44.1        | 55.2        | 69.4        |
| Lifted [25]                         | 56.8        | 22.9           | 47.8        | 60.1        | 71.4        | 81.9        |
| Angular [33]                        | 58.3        | 27.8           | 50.6        | 64.1        | 74.1        | 83.2        |
| Triplet <sub>2</sub> +DWS [37]      | 58.0        | 24.8           | 49.8        | 61.6        | 73.2        | 83.6        |
| <b>DVML+Triplet<sub>2</sub>+DWS</b> | <b>61.4</b> | <b>28.8</b>    | <b>52.7</b> | <b>65.1</b> | <b>75.5</b> | <b>84.3</b> |
| HDC [40]                            | -           | -              | 53.6        | 65.7        | 77.0        | 85.6        |
| Proxy-NCA [16]                      | 59.5        | -              | 49.2        | 61.9        | 67.9        | 72.4        |

GoogLeNet’s last average pooling layer as the reconstruction target. We use two fully-connected layers with output dimension 512 and 1024 respectively as the decoder network, where tanh is used as the activation function. We randomly initialize all the added fully-connected layers.

There are two phases in the training procedure. In the first phase, we cut off the back-propagation of the gradients from the decoder network for the stability of the embedding part. We empirically set  $\lambda_1 = 1, \lambda_2 = 1, \lambda_3 = 0.1$ , and  $\lambda_4 = 1$ . In the second phase, we release the constraint and empirically set  $\lambda_1 = 0.8, \lambda_2 = 1, \lambda_3 = 0.2$ , and  $\lambda_4 = 0.8$ . As the experimental study in [25] showed that the embedding size does not largely affect the performance, we follow [33] and fix the embedding size to 512 in all the experiments. We set the batch size as 128 for the pair-based and batch-based input and 120 for the triplet input. For the iterations of discriminative sample generation, we set  $T = 20$  throughout the experiments. To optimize the objective, we take Adam [11] as the optimizer and set the training rate to be 0.0001.

## 4 Experiments

To demonstrate the effectiveness of our DVML, we conduct experiments on three widely-used datasets for both retrieval and clustering tasks.

### 4.1 Settings

We follow [25,24,33] to split the training and testing set in a zero-shot manner for all the datasets.

**Table 4.** Average p-value and relative deviation (%) on three datasets

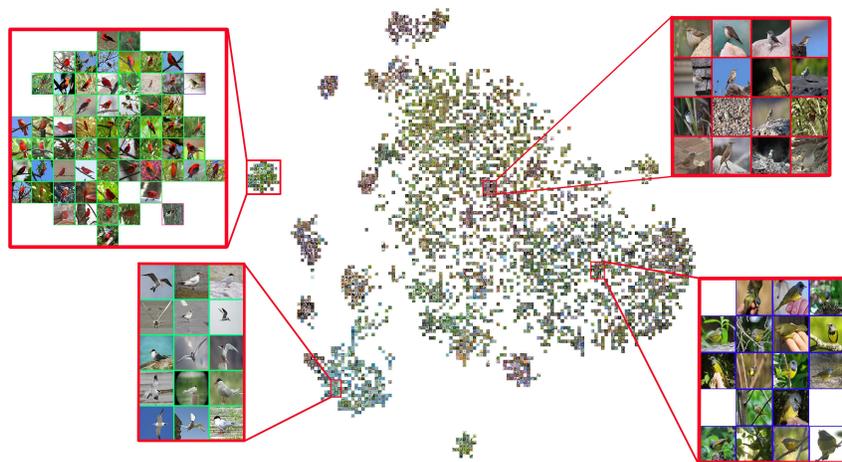
| Train               | Cars196      | CUB-200-2011 | Products     |
|---------------------|--------------|--------------|--------------|
| Triplet p-value     | 76.00 ± 1.25 | 76.87 ± 1.42 | 83.87 ± 8.24 |
| Triplet             | 79.99        | 86.14        | 73.10        |
| <b>DVML+Triplet</b> | <b>40.75</b> | <b>44.50</b> | <b>44.48</b> |
| Test                | Cars196      | CUB-200-2011 | Products     |
| Triplet p-value     | 74.87 ± 2.29 | 77.01 ± 1.19 | 83.75 ± 8.12 |
| Triplet             | 97.98        | 105.80       | 73.46        |
| <b>DVML+Triplet</b> | <b>57.08</b> | <b>57.56</b> | <b>46.94</b> |

- The CUB-200-2011 dataset [31] contains 11,788 images from 200 bird species. We take the first 100 classes with 5,864 images for training, and the rest 100 classes with 5,924 images for testing.
- The Cars196 dataset [13] contains 16,185 images of 196 car types. We take the first 98 classes with 8,054 images for training, and the rest 98 classes with 8,131 images for testing.
- The Stanford Online Products dataset [25] contains 120,053 images of 22,634 products. We take the first 11,318 classes with 59,551 images for training, and the rest 11,316 classes with 60,502 images for testing.

In the retrieval task, we calculate the percentage of test samples that have at least one sample from the same class in  $R$  nearest neighbors. In the clustering task, we report the NMI [25] score and  $F_1$  [25] score. For NMI, the input is a set of clusters  $\Omega = \{\omega_1, \dots, \omega_K\}$  and the ground truth classes  $\mathbb{C} = \{c_1, \dots, c_K\}$ .  $\omega_i$  indicates the samples that are assigned to the  $i$ th cluster, and  $c_j$  is the set of samples with the ground truth label  $j$ . NMI is the ratio of mutual information and the mean entropy of clusters and the ground truth:  $\text{NMI}(\Omega, \mathbb{C}) = \frac{2I(\Omega; \mathbb{C})}{(H(\Omega) + H(\mathbb{C}))}$ .  $F_1$  score is defined as the harmonic mean of precision and recall:  $F_1 = \frac{2PR}{P+R}$ .

## 4.2 Compared Methods

We apply our deep variational metric learning framework to three aforementioned baseline methods. They are Triplet loss [36], N-pair loss [23], and Triplet<sub>2</sub> loss with Distance Weighted Sampling [37]. We compare the performance of baseline methods before and after using our DVML framework to demonstrate the effectiveness of our proposed framework. We also compare DVML with other widely used or state-of-the-art methods, where there are two categories: designing sampling algorithms and modifying loss functions. For loss functions, we compare our methods with the widely-used Contrastive loss [7], the Lifted-Structure loss [25], and the state-of-the-art Angular loss [33]. For sampling methods, we



**Fig. 3.** Visualization of the proposed DVML+N-pair with Barnes-Hut t-SNE [30] on the CUB-200-2011 test set. The color of the bounding box for each image represents the label. (Best viewed when zoomed in.)

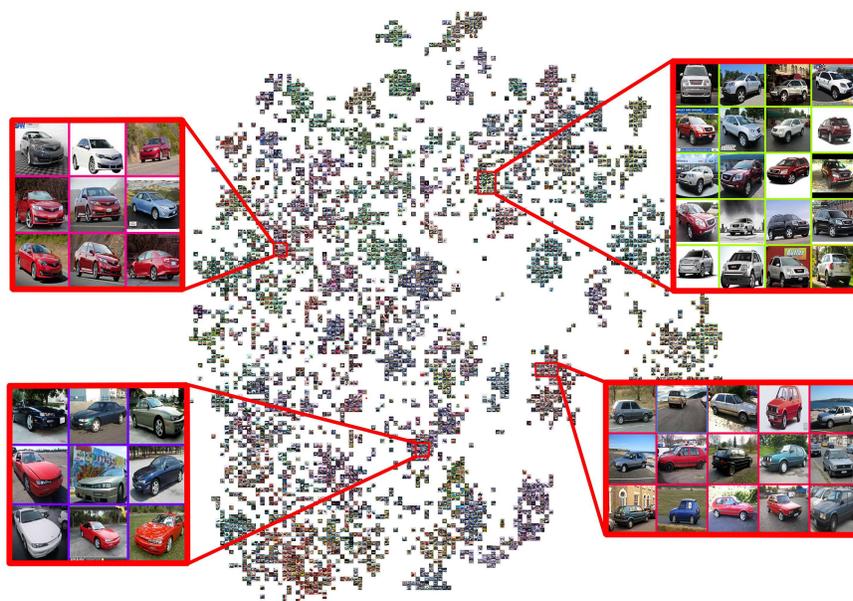
take the state-of-the-art methods including HDC [40] and an upper bound generating method Proxy-NCA [16]. We report the most relevant results according to their original papers. Except them, we re-implement all the compared methods. In the re-implementation, we observe some differences from the reported results in original papers, but this does not affect the fairness of comparison.

### 4.3 Quantitative Results

Table 1, Table 2, and Table 3 present the experimental results of our DVML and all compared methods on the Cars196, Stanford Online Product, and CUB-200-2011 datasets respectively.

From the comparison with baseline method, we notice that our proposed DVML significantly improved the performance of baseline methods. It is surprising that our proposed DVML significantly improves the performance of N-pair loss which has already gained success on the Cars196 and Stanford Online Products datasets, which further proves the limitation we stress before does exist. In the CUB-200-2011, our proposed DVML’s effectiveness is relatively less significant than the other two datasets. We suppose that it is due to the different nature of datasets. In Cars196 and Stanford Online Products, the difficulty lies in a large range of intra-class variance, while in CUB-200-2011, the difficulty lies in localizing discriminative fine-grained regions.

In the comparison with other methods, we observe that on the Cars196 dataset, our **DVML+Triplet<sub>2</sub>+DWS** achieves better performance than the previous state-of-the-art. In the other two datasets, our DVML also achieves



**Fig. 4.** Visualization of the proposed DVML+N-pair with Barnes-Hut t-SNE [30] on the Cars196 test set. The color of the bounding box for each image represents the label. (Best viewed when zoomed in.)

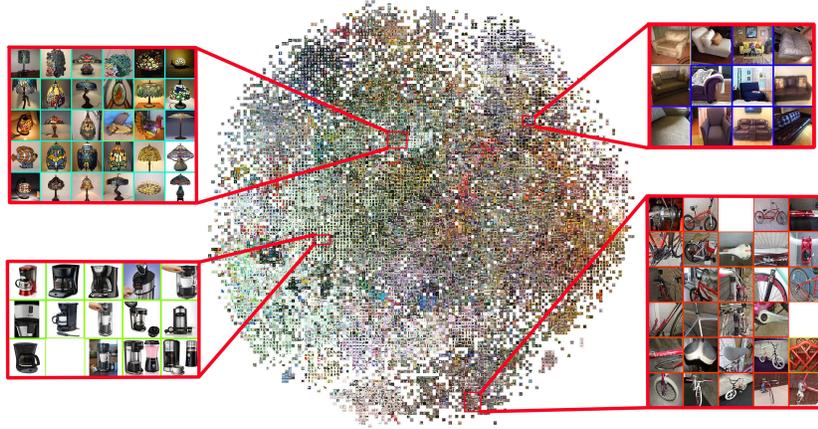
comparable performance. It is noticeable that we take both sampling algorithms and modified loss functions as baselines and compete with the state-of-the-art in both of the categories, which further shows the effectiveness of our DVML.

To further verify our assumption, we first apply Kolmogorov-Smirnov test [22] to the central features learned with Triplet loss [36,18] and measure the p-value. The results in Table 4 show that central features of classes in every dataset probably obey the isotropic Gaussian and their distributions are probably similar because the deviation of p-value over classes is small. We also measure the average relative deviation of features with and without DVML, and the result suggests that our DVML does remove intra-class variance from output features and helps to explicitly learn class centers.

#### 4.4 Qualitative Results

Using a well-known visualization method t-SNE [30], we first visualize the central latent space of features learned with the N-pair loss [23] to illustrate our insight in Fig. 1.

Fig. 3, Fig. 4, and Fig. 5 show the visualization of DVML+N-pair on the CUB-200-2011, Cars196 and Stanford Online Products datasets. The figures are best viewed when zoomed in. The color of the bounding box on each samples' images represent their class label. Following [25,33], we enlarge certain regions



**Fig. 5.** Visualization of the proposed DVML+N-pair with Barnes-Hut t-SNE [30] on the Stanford Online Products test set. The color of the bounding box for each image represents the label. (Best viewed when zoomed in.)

to highlight the discriminability of learned features. The visualization explicitly show that our proposed DVML learns a good metric which well preserves the distance relationships among classes, given a large range of intra-class variance.

## 5 Conclusion

In this paper, we have presented a novel applicable framework: deep variational metric learning (DVML). We assume and illustrate that the distribution of intra-class variance is invariant among classes. To the best of our knowledge, this is the first work to disentangle intra-class variance via variational inference, and the first to leverage the intra-class variance’s distribution to generate discriminative samples. We stress that with our DVML, current metric learning algorithms could be significantly improved. Furthermore, there are many future works, including image generating given certain classes, and utilizing those generated images to further improve robustness of metric learning models.

## 6 Acknowledgement

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFA0700802, in part by the National Natural Science Foundation of China under Grant 61672306, Grant U1713214, Grant 61572271, and in part by the Shenzhen Fundamental Research Fund (Subject Arrangement) under Grant JCYJ20170412170602564.

## References

1. Davis, J.V., Kulis, B., Jain, P., Sra, S., Dhillon, I.S.: Information-theoretic metric learning. In: ICML. pp. 209–216 (2007) [3](#)
2. Duan, Y., Wang, Z., Lu, J., Lin, X., Zhou, J.: Graphbit: Bitwise interaction mining via deep reinforcement learning. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018) [3](#)
3. Duan, Y., Zheng, W., Lin, X., Lu, J., Zhou, J.: Deep adversarial metric learning. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018) [1](#), [3](#)
4. Fragoso, V., Ramanan, D.: Bayesian embeddings for long-tailed datasets (2018) [3](#), [4](#)
5. Globerson, A., Roweis, S.T.: Metric learning by collapsing classes. In: NIPS. pp. 451–458 (2006) [3](#)
6. Guillaumin, M., Verbeek, J., Schmid, C.: Is that you? Metric learning approaches for face identification. In: CVPR. pp. 498–505 (2009) [3](#)
7. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: CVPR. pp. 1735–1742 (2006) [1](#), [3](#), [8](#), [9](#), [10](#), [11](#)
8. Harwood, B., Carneiro, G., Reid, I., Drummond, T.: Smart mining for deep metric learning. In: ICCV. pp. 2821–2829 (2017) [1](#), [3](#)
9. Hu, J., Lu, J., Tan, Y.P.: Deep metric learning for visual tracking. TCSVT **26**(11), 2056–2068 (2016) [3](#)
10. Hu, J., Lu, J., Tan, Y.P.: Discriminative deep metric learning for face and kinship verification. TIP **26**(9), 4269–4282 (2017) [1](#), [3](#)
11. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014) [10](#)
12. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013) [3](#), [6](#)
13. Krause, J., Stark, M., Deng, J., Feifei, L.: 3d object representations for fine-grained categorization. In: ICCVW (2013) [11](#)
14. Larsen, A.B.L., Sønderby, S.K., Larochelle, H., Winther, O.: Autoencoding beyond pixels using a learned similarity metric. arXiv preprint arXiv:1512.09300 (2015) [3](#)
15. Liu, Z., Wang, D., Lu, H.: Stepwise metric promotion for unsupervised video person re-identification. In: ICCV. pp. 2429–2438 (2017) [3](#)
16. Movshovitz-Attias, Y., Toshev, A., Leung, T.K., Ioffe, S., Singh, S.: No fuss distance metric learning using proxies. In: ICCV. pp. 360–368 (2017) [3](#), [8](#), [9](#), [10](#), [12](#)
17. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) **115**(3), 211–252 (2015). <https://doi.org/10.1007/s11263-015-0816-y> [9](#)
18. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: CVPR. pp. 815–823 (2015) [1](#), [3](#), [5](#), [8](#), [9](#), [10](#), [13](#)
19. Schultz, M., Joachims, T.: Learning a distance metric from relative comparisons. In: NIPS. pp. 41–48 (2004) [3](#)
20. Shalev-Shwartz, S., Singer, Y., Ng, A.Y.: Online and batch learning of pseudo-metrics. In: ICML (2004) [3](#)
21. Shrivastava, A., Gupta, A., Girshick, R.: Training region-based object detectors with online hard example mining. In: CVPR. pp. 761–769 (2016) [1](#), [3](#)

22. Smirnov, N.: Table for estimating the goodness of fit of empirical distributions. *Ann. Math. Statist.* **19**(2), 279–281 (06 1948). <https://doi.org/10.1214/aoms/1177730256>, <https://doi.org/10.1214/aoms/1177730256> **13**
23. Sohn, K.: Improved deep metric learning with multi-class n-pair loss objective. In: NIPS. pp. 1849–1857 (2016) **1, 2, 3, 5, 8, 9, 10, 11, 13**
24. Song, H.O., Jegelka, S., Rathod, V., Murphy, K.: Deep metric learning via facility location. In: CVPR. pp. 5382–5390 (2017) **1, 3, 10**
25. Song, H.O., Xiang, Y., Jegelka, S., Savarese, S.: Deep metric learning via lifted structured feature embedding. In: CVPR. pp. 4004–4012 (2016) **8, 9, 10, 11, 13**
26. Sun, C., Wang, D., Lu, H.: Person re-identification via distance metric learning with latent variables. *IEEE Transactions on Image Processing* **26**(1), 23–34 (2017) **3**
27. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S.E., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: CVPR. pp. 1–9 (2015) **9**
28. Tokui, S., Oono, K., Hido, S., Clayton, J.: Chainer: a next-generation open source framework for deep learning. In: Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Twenty-ninth Annual Conference on Neural Information Processing Systems (NIPS) (2015), [http://learningsys.org/papers/LearningSys\\_2015\\_paper\\_33.pdf](http://learningsys.org/papers/LearningSys_2015_paper_33.pdf) **9**
29. Tolstikhin, I., Bousquet, O., Gelly, S., Schoelkopf, B.: Wasserstein auto-encoders. arXiv preprint arXiv:1711.01558 (2017) **3**
30. Van Der Maaten, L.: Accelerating t-sne using tree-based algorithms. *JMLR* **15**(1), 3221–3245 (2014) **2, 12, 13, 14**
31. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.J.: The Caltech-UCSD Birds-200-2011 dataset. Tech. Rep. CNS-TR-2011-001, California Institute of Technology (2011) **11**
32. Wang, D., Tan, X.: Robust distance metric learning via bayesian inference. *IEEE Transactions on Image Processing* **27**(3), 1542–1553 (2018) **3**
33. Wang, J., Zhou, F., Wen, S., Liu, X., Lin, Y.: Deep metric learning with angular loss. In: ICCV. pp. 2593–2601 (2017) **1, 3, 8, 9, 10, 11, 13**
34. Wang, X., Gupta, A.: Unsupervised learning of visual representations using videos. In: ICCV. pp. 2794–2802 (2015) **1, 3**
35. Wang, X., Hua, G., Han, T.X.: Discriminative tracking by metric learning. In: ECCV. pp. 200–214 (2010) **3**
36. Weinberger, K.Q., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. *JMLR* **10**(Feb), 207–244 (2009) **1, 3, 5, 8, 9, 10, 11, 13**
37. Wu, C.Y., Manmatha, R., Smola, A.J., Krähenbühl, P.: Sampling matters in deep embedding learning. arXiv preprint arXiv:1706.07567 (2017) **1, 3, 6, 8, 9, 10, 11**
38. Yang, L., Jin, R., Sukthankar, R.: Bayesian active distance metric learning. arXiv preprint arXiv:1206.5283 (2012) **3**
39. Yu, H.X., Wu, A., Zheng, W.S.: Cross-view asymmetric metric learning for unsupervised person re-identification. In: ICCV. pp. 994–1002 (2017) **3**
40. Yuan, Y., Yang, K., Zhang, C.: Hard-aware deeply cascaded embedding. In: ICCV. pp. 814–823 (2017) **1, 3, 8, 9, 10, 12**
41. Zhou, J., Yu, P., Tang, W., Wu, Y.: Efficient online local metric adaptation via negative samples for person re-identification. In: ICCV. pp. 2420–2428 (2017) **3**