

SoftTriple Loss: Deep Metric Learning Without Triplet Sampling

Qi Qian¹ Lei Shang² Baigui Sun² Juhua Hu³ Hao Li² Rong Jin¹

¹ Alibaba Group, Bellevue, WA, 98004, USA

² Alibaba Group, Hangzhou, China

³ School of Engineering and Technology

University of Washington, Tacoma, WA, 98402, USA

{qi.qian, sl172005, baigui.sbg, lihao.lh, jinrong.jr}@alibaba-inc.com, juhuah@uw.edu

Abstract

Distance metric learning (DML) is to learn the embeddings where examples from the same class are closer than examples from different classes. It can be cast as an optimization problem with triplet constraints. Due to the vast number of triplet constraints, a sampling strategy is essential for DML. With the tremendous success of deep learning in classifications, it has been applied for DML. When learning embeddings with deep neural networks (DNNs), only a mini-batch of data is available at each iteration. The set of triplet constraints has to be sampled within the mini-batch. Since a mini-batch cannot capture the neighbors in the original set well, it makes the learned embeddings sub-optimal. On the contrary, optimizing SoftMax loss, which is a classification loss, with DNN shows a superior performance in certain DML tasks. It inspires us to investigate the formulation of SoftMax. Our analysis shows that SoftMax loss is equivalent to a smoothed triplet loss where each class has a single center. In real-world data, one class can contain several local clusters rather than a single one, e.g., birds of different poses. Therefore, we propose the SoftTriple loss to extend the SoftMax loss with multiple centers for each class. Compared with conventional deep metric learning algorithms, optimizing SoftTriple loss can learn the embeddings without the sampling phase by mildly increasing the size of the last fully connected layer. Experiments on the benchmark fine-grained data sets demonstrate the effectiveness of the proposed loss function. Code is available at <https://github.com/idstcv/SoftTriple>.

1. Introduction

Distance metric learning (DML) has been extensively studied in the past decades due to its broad range of applications, e.g., k -nearest neighbor classification [29], image retrieval [24] and clustering [31]. With an appropriate

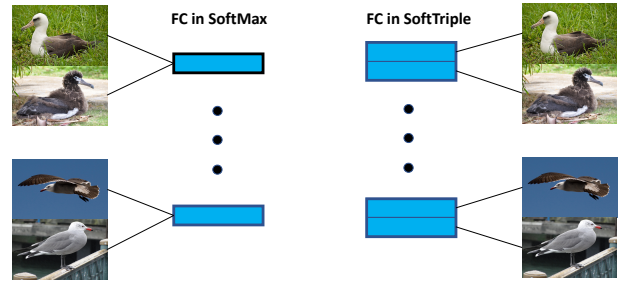


Figure 1. Illustration of the proposed SoftTriple loss. In conventional SoftMax loss, each class has a representative center in the last fully connected layer. Examples in the same class will be collapsed to the same center. It may be inappropriate for the real-world data as illustrated. In contrast, SoftTriple loss keeps multiple centers (e.g., 2 centers per class in this example) in the fully connected layer and each image will be assigned to one of them. It is more flexible for modeling intra-class variance in real-world data sets.

distance metric, examples from the same class should be closer than examples from different classes. Many algorithms have been proposed to learn a good distance metric [15, 16, 21, 29].

In most of conventional DML methods, examples are represented by hand-crafted features, and DML is to learn a feature mapping to project examples from the original feature space to a new space. The distance can be computed as the Mahalanobis distance [11]

$$\text{dist}_M(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^\top M (\mathbf{x}_i - \mathbf{x}_j)$$

where M is the learned distance metric. With this formulation, the main challenge of DML is from the dimensionality of input space. As a metric, the learned matrix M has to be positive semi-definite (PSD) while the cost of keeping the matrix PSD can be up to $\mathcal{O}(d^3)$, where d is the dimensionality of original features. The early work directly applies PCA to shrink the original space [29]. Later, various strategies are developed to reduce the computational cost [16, 17].

Those approaches can obtain the good metric from the input features, but the hand-crafted features are task independent and may cause the loss of information, which limits the performance of DML. With the success of deep neural networks in classification [7], researchers consider to learn the embeddings directly from deep neural networks [15, 21]. Without the explicit feature extraction, deep metric learning boosts the performance by a large margin [21]. In deep metric learning, the dimensionality of input features is no longer a challenge since neural networks can learn low-dimensional features directly from raw materials, e.g., images, documents, etc. In contrast, generating appropriate constraints for optimization becomes challenging for deep metric learning.

It is because most of deep neural networks are trained with the stochastic gradient descent (SGD) algorithm and only a mini-batch of examples are available at each iteration. Since embeddings are optimized with the loss defined on an anchor example and its neighbors (e.g., the active set of pairwise [31] or triplet [29] constraints), the examples in a mini-batch may not be able to capture the overall neighborhood well, especially for relatively large data sets. Moreover, a mini-batch contains $\mathcal{O}(m^2)$ pairs and $\mathcal{O}(m^3)$ triplets, where m is the size of the mini-batch. An effective sampling strategy over the mini-batch is essential even for a small batch (e.g., 32) to learn the embeddings efficiently. Many efforts have been devoted to studying sampling an informative mini-batch [19, 21] and sampling triplets within a mini-batch [12, 24]. Some work also tried to reduce the total number of triplets with proxies [14, 18]. The sampling phase for the mini-batch and constraints not only loses the information but also makes the optimization complicated. In this work, we consider to learn embeddings without constraints sampling.

Recently, researches have shown that embeddings obtained directly from optimizing SoftMax loss, which is proposed for classification, perform well on the simple distance based tasks [22, 30] and face recognition [2, 9, 10, 27, 28]. It inspires us to investigate the formulation of SoftMax loss. Our Analysis demonstrates that SoftMax loss is equivalent to a smoothed triplet loss. By providing a single center for each class in the last fully connected layer, the triplet constraint derived by SoftMax loss can be defined on an original example, its corresponding center and a center from a different class. Therefore, embeddings obtained by optimizing SoftMax loss can work well as a distance metric. However, a class in real-world data can consist of multiple local clusters as illustrated in Fig. 1 and a single center is insufficient to capture the inherent structure of the data. Consequently, embeddings learned from SoftMax loss can fail in the complex scenario [22].

In this work, we propose to improve SoftMax loss by introducing multiple centers for each class and the novel

loss is denoted as SoftTriple loss. Compared with a single center, multiple ones can capture the hidden distribution of the data better due to the fact that they help to reduce the intra-class variance. This property is also crucial to reserve the triplet constraints over original examples while training with multiple centers. Compared with existing deep DML methods, the number of triplets in SoftTriple is linear in the number of original examples. Since the centers are encoded in the last fully connected layer, SoftTriple loss can be optimized without sampling triplets. Fig. 1 illustrates the proposed SoftTriple loss. Apparently, SoftTriple loss has to determine the number of centers for each class. To alleviate this issue, we develop a strategy that sets a sufficiently large number of centers for each class at the beginning and then applies $L_{2,1}$ norm to obtain a compact set of centers. We demonstrate the proposed loss on the fine-grained visual categorization tasks, where capturing local clusters is essential for good performance [17].

The rest of this paper is organized as follows. Section 2 reviews the related work of conventional distance metric learning and deep metric learning. Section 3 analyzes the SoftMax loss and proposes the SoftTriple loss accordingly. Section 4 conducts comparisons on benchmark data sets. Finally, Section 5 concludes this work and discusses future directions.

2. Related Work

Distance metric learning Many DML methods have been developed when input features are provided [29, 31]. The dimensionality of input features is a critical challenge for those methods due to the PSD projection, and many strategies have been proposed to alleviate it. The most straightforward way is to reduce the dimension of input space by PCA [29]. However, PCA is task independent and may hurt the performance of learned embeddings. Some works try to reduce the number of valid parameters with the low-rank assumption [8]. [16] decreases the computational cost by reducing the number of PSD projections. [17] proposes to learn the dual variables in the low-dimensional space introduced by random projections and then recover the metric in the original space. After addressing the challenge from the dimensionality, the hand-crafted features become the bottleneck of performance improvement.

The forms of constraints for metric learning are also developed in these methods. Early work focuses on optimizing pairwise constraints, which require the distances between examples from the same class small while those from different classes large [31]. Later, [29] develops the triplet constraints, where given an anchor example, the distance between the anchor point and a similar example should be smaller than that between the anchor point and a dissimilar example by a large margin. It is obvious that the number of pairwise constraints is $\mathcal{O}(n^2)$ while that of triplet

constraints can be up to $\mathcal{O}(n^3)$, where n is the number of original examples. Compared with the pairwise constraints, triplet constraints optimize the geometry of local cluster and are more applicable for modeling intra-class variance. In this work, we will focus on the triplet constraints.

Deep metric learning Deep metric learning aims to learn the embeddings directly from the raw materials (e.g., images) by deep neural networks [15, 21]. With the task dependent embeddings, the performance of metric learning has a dramatical improvement. However, most of deep models are trained with SGD that allows only a mini-batch of data at each iteration. Since the size of mini-batch is small, the information in it is limited compared to the original data. To alleviate this problem, algorithms have to develop an effective sampling strategy to generate the mini-batch and then sample triplet constraints from it. A straightforward way is increasing the size of mini-batch [21]. However, the large mini-batch will suffer from the GPU memory limitation and can also increase the challenge of sampling triplets. Later, [19] proposes to generate the mini-batch from neighbor classes. Besides, there are various sampling strategies for obtaining constraints [3, 12, 21, 24]. [21] proposes to sample the semi-hard negative examples. [24] adopts all negative examples within the margin for each positive pair. [12] develops distance weighted sampling that samples examples according to the distance from the anchor example. [3] selects hard triplets with a dynamic violate margin from a hierarchical class-level tree. However, all of these strategies may fail to capture the distribution of the whole data set. Moreover, they make the optimization in deep DML complicated.

Learning with proxies Recently, some researchers consider to reduce the total number of triplets to alleviate the challenge from the large number of triplets. [14] constructs the triplet loss with one original example and two proxies. Since the number of proxies is significantly less than the number of original examples, proxies can be kept in the memory that help to avoid the sampling over different batches. However, it only provides a single proxy for each class when label information is available, which is similar to SoftMax. [18] proposes a conventional DML algorithm to construct the triplet loss only with latent examples, which assigns multiple centers for each class and further reduces the number of triplets. In this work, we propose to learn the embeddings by optimizing the proposed SoftTriple loss to eliminate the sampling phase and capture the local geometry of each class simultaneously.

3. SoftTriple Loss

In this section, we first introduce the SoftMax loss and the triplet loss and then study the relationship between them to derive the SoftTriple loss.

Denote the embedding of the i -th example as \mathbf{x}_i and the corresponding label as y_i , then the conditional probability output by a deep neural network can be estimated via the SoftMax operator

$$\Pr(Y = y_i | \mathbf{x}_i) = \frac{\exp(\mathbf{w}_{y_i}^\top \mathbf{x}_i)}{\sum_j^C \exp(\mathbf{w}_j^\top \mathbf{x}_i)}$$

where $[\mathbf{w}_1, \dots, \mathbf{w}_C] \in \mathbb{R}^{d \times C}$ is the last fully connected layer. C denotes the number of classes and d is the dimension of embeddings. The corresponding SoftMax loss is

$$\ell_{\text{SoftMax}}(\mathbf{x}_i) = -\log \frac{\exp(\mathbf{w}_{y_i}^\top \mathbf{x}_i)}{\sum_j \exp(\mathbf{w}_j^\top \mathbf{x}_i)}$$

A deep model can be learned by minimizing losses over examples. This loss has been prevalently applied for classification task [7].

Given a triplet $(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k)$, DML aims to learn good embeddings such that examples from the same class are closer than examples from different classes, i.e.,

$$\forall i, j, k, \quad \|\mathbf{x}_i - \mathbf{x}_k\|_2^2 - \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 \geq \delta$$

where \mathbf{x}_i and \mathbf{x}_j are from the same class and \mathbf{x}_k is from a different class. δ is a predefined margin. When each example has the unit length (i.e., $\|\mathbf{x}\|_2 = 1$), the triplet constraint can be simplified as

$$\forall i, j, k, \quad \mathbf{x}_i^\top \mathbf{x}_j - \mathbf{x}_i^\top \mathbf{x}_k \geq \delta \quad (1)$$

where we ignore the rescaling of δ . The corresponding triplet loss can be written as

$$\ell_{\text{triplet}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) = [\delta + \mathbf{x}_i^\top \mathbf{x}_k - \mathbf{x}_i^\top \mathbf{x}_j]_+ \quad (2)$$

It is obvious from Eqn. 1 that the number of total triplets can be cubic in the number of examples, which makes sampling inevitable for most of triplet based DML algorithms.

With the unit length for both \mathbf{w} and \mathbf{x} , the normalized SoftMax loss can be written as

$$\ell_{\text{SoftMax}_{\text{norm}}}(\mathbf{x}_i) = -\log \frac{\exp(\lambda \mathbf{w}_{y_i}^\top \mathbf{x}_i)}{\sum_j \exp(\lambda \mathbf{w}_j^\top \mathbf{x}_i)} \quad (3)$$

where λ is a scaling factor.

Surprisingly, we find that minimizing the normalized SoftMax loss with the smooth term λ is equivalent to optimizing a smoothed triplet loss.

Proposition 1.

$$\ell_{\text{SoftMax}_{\text{norm}}}(\mathbf{x}_i) = \max_{\mathbf{p} \in \Delta} \lambda \sum_j \mathbf{p}_j \mathbf{x}_i^\top (\mathbf{w}_j - \mathbf{w}_{y_i}) + H(\mathbf{p}) \quad (4)$$

where $\mathbf{p} \in \mathbb{R}^C$ is a distribution over classes and Δ is the simplex as $\Delta = \{\mathbf{p} | \sum_j \mathbf{p}_j = 1, \forall j, \mathbf{p}_j \geq 0\}$. $H(\mathbf{p})$ denotes the entropy of the distribution \mathbf{p} .

Proof. According to the K.K.T. condition [1], the distribution \mathbf{p} in Eqn. 4 has the closed-form solution

$$\mathbf{p}_j = \frac{\exp(\lambda \mathbf{x}_i^\top (\mathbf{w}_j - \mathbf{w}_{y_i}))}{\sum_j \exp(\lambda \mathbf{x}_i^\top (\mathbf{w}_j - \mathbf{w}_{y_i}))}$$

Therefore, we have

$$\begin{aligned} \ell_{\text{SoftMax}_{\text{norm}}}(\mathbf{x}_i) &= \lambda \sum_j \mathbf{p}_j \mathbf{x}_i^\top (\mathbf{w}_j - \mathbf{w}_{y_i}) + H(\mathbf{p}) \\ &= \log\left(\sum_j \exp(\lambda \mathbf{x}_i^\top (\mathbf{w}_j - \mathbf{w}_{y_i}))\right) = -\log \frac{\exp(\lambda \mathbf{w}_{y_i}^\top \mathbf{x}_i)}{\sum_j \exp(\lambda \mathbf{w}_j^\top \mathbf{x}_i)} \end{aligned}$$

□

Remark 1 Proposition 1 indicates that the SoftMax loss optimizes the triplet constraints consisting of an original example and two centers, i.e., $(\mathbf{x}_i, \mathbf{w}_{y_i}, \mathbf{w}_j)$. Compared with triplet constraints in Eqn. 1, the target of SoftMax loss is

$$\forall i, j, \quad \mathbf{x}_i^\top \mathbf{w}_{y_i} - \mathbf{x}_i^\top \mathbf{w}_j \geq 0$$

Consequently, the embeddings learned by minimizing SoftMax loss can be applicable for the distance-based tasks while it is designed for the classification task.

Remark 2 Without the entropy regularizer, the loss becomes

$$\max_{\mathbf{p} \in \Delta} \lambda \sum_j \mathbf{p}_j \mathbf{x}_i^\top \mathbf{w}_j - \lambda \mathbf{x}_i^\top \mathbf{w}_{y_i}$$

which is equivalent to

$$\max_j \{\mathbf{x}_i^\top \mathbf{w}_j\} - \mathbf{x}_i^\top \mathbf{w}_{y_i}$$

Explicitly, it punishes the triplet with the most violation and becomes zero when the nearest neighbor of \mathbf{x}_i is the corresponding center \mathbf{w}_{y_i} . The entropy regularizer reduces the influence from outliers and makes the loss more robust. λ trades between the hardness of triplets and the regularizer. Moreover, minimizing the maximal entropy can make the distribution concentrated and further push the example away from irrelevant centers, which implies a large margin property.

Remark 3 Applying the similar analysis to the ProxyNCA loss [14]: $\ell_{\text{ProxyNCA}}(\mathbf{x}_i) = -\log \frac{\exp(\mathbf{w}_{y_i}^\top \mathbf{x}_i)}{\sum_{j \neq y_i} \exp(\mathbf{w}_j^\top \mathbf{x}_i)}$, we have

$$\ell_{\text{ProxyNCA}}(\mathbf{x}_i) = \max_{\mathbf{p} \in \Delta} \lambda \sum_{j \neq y_i} \mathbf{p}_j \mathbf{x}_i^\top (\mathbf{w}_j - \mathbf{w}_{y_i}) + H(\mathbf{p})$$

where $\mathbf{p} \in \mathbb{R}^{C-1}$. Compared with the SoftMax loss, it eliminates the benchmark triplet containing only the corresponding class center, which makes the loss unbounded. Our analysis suggests that the loss can be bounded as in Eqn. 2: $\ell_{\text{ProxyNCA}}^{\text{hinge}}(\mathbf{x}_i) = [-\log \frac{\exp(\mathbf{w}_{y_i}^\top \mathbf{x}_i)}{\sum_{j \neq y_i} \exp(\mathbf{w}_j^\top \mathbf{x}_i)}]_+$. Validating the bounded loss is out of the scope of this work.

Despite optimizing SoftMax loss can learn the meaningful feature embeddings, the drawback is straightforward. It assumes that there is only a single center for each class while a real-world class can contain multiple local clusters due to the large intra-class variance as in Fig. 1. The triplet constraints generated by conventional SoftMax loss is too brief to capture the complex geometry of the original data. Therefore, we introduce multiple centers for each class.

3.1. Multiple Centers

Now, we assume that each class has K centers. Then, the similarity between the example \mathbf{x}_i and the class c can be defined as

$$\mathcal{S}_{i,c} = \max_k \mathbf{x}_i^\top \mathbf{w}_c^k \quad (5)$$

Note that other definitions of similarity can be applicable for this scenario (e.g., $\min_{\mathbf{z} \in \mathbb{R}^K} \|[\mathbf{w}_c^1, \dots, \mathbf{w}_c^K] \mathbf{z} - \mathbf{x}_i\|_2$). We adopt a simple form to illustrate the influence of multiple centers.

With the definition of the similarity, the triplet constraint requires an example to be closer to its corresponding class than other classes

$$\forall j, \quad \mathcal{S}_{i,y_i} - \mathcal{S}_{i,j} \geq 0$$

As we mentioned above, minimizing the entropy term $H(\mathbf{p})$ can help to pull the example to the corresponding center. To break the tie explicitly, we consider to introduce a small margin as in the conventional triplet loss in Eqn. 1 and define the constraints as

$$\forall j_{j \neq y_i}, \quad \mathcal{S}_{i,y_i} - \mathcal{S}_{i,j} \geq \delta$$

By replacing the similarity in Eqn. 4, we can obtain the

HardTriple loss as

$$\begin{aligned} \ell_{\text{HardTriple}}(\mathbf{x}_i) &= \max_{\mathbf{p} \in \Delta} \lambda \left(\sum_{j \neq y_i} \mathbf{p}_j (\mathcal{S}_{i,j} - (\mathcal{S}_{i,y_i} - \delta)) \right. \\ &\quad \left. + \mathbf{p}_{y_i} (\mathcal{S}_{i,y_i} - \delta - (\mathcal{S}_{i,y_i} - \delta)) \right) + H(\mathbf{p}) \\ &= -\log \frac{\exp(\lambda(\mathcal{S}_{i,y_i} - \delta))}{\exp(\lambda(\mathcal{S}_{i,y_i} - \delta)) + \sum_{j \neq y_i} \exp(\lambda \mathcal{S}_{i,j})} \end{aligned} \quad (6)$$

HardTriple loss improves the SoftMax loss by providing multiple centers for each class. However, it requires the max operator to obtain the nearest center in each class while this operator is not smooth and the assignment can be sensitive between multiple centers. Inspired by the SoftMax loss, we can improve the robustness by smoothing the max operator.

Consider the problem

$$\max_k \mathbf{x}_i^\top \mathbf{w}_c^k$$

which is equivalent to

$$\max_{\mathbf{q} \in \Delta} \sum_k \mathbf{q}_k \mathbf{x}_i^\top \mathbf{w}_c^k \quad (7)$$

we add the entropy regularizer to the distribution \mathbf{q} as

$$\max_{\mathbf{q} \in \Delta} \sum_k \mathbf{q}_k \mathbf{x}_i^\top \mathbf{w}_c^k + \gamma H(\mathbf{q})$$

With a similar analysis as in Proposition 1, \mathbf{q} has the closed-form solution as

$$\mathbf{q}_k = \frac{\exp(\frac{1}{\gamma} \mathbf{x}_i^\top \mathbf{w}_c^k)}{\sum_k \exp(\frac{1}{\gamma} \mathbf{x}_i^\top \mathbf{w}_c^k)}$$

Taking it back to the Eqn. 7, we define the relaxed similarity between the example \mathbf{x}_i and the class c as

$$\mathcal{S}'_{i,c} = \sum_k \frac{\exp(\frac{1}{\gamma} \mathbf{x}_i^\top \mathbf{w}_c^k)}{\sum_k \exp(\frac{1}{\gamma} \mathbf{x}_i^\top \mathbf{w}_c^k)} \mathbf{x}_i^\top \mathbf{w}_c^k$$

By applying the smoothed similarity, we define the SoftTriple loss as

$$\begin{aligned} \ell_{\text{SoftTriple}}(\mathbf{x}_i) &= -\log \frac{\exp(\lambda(\mathcal{S}'_{i,y_i} - \delta))}{\exp(\lambda(\mathcal{S}'_{i,y_i} - \delta)) + \sum_{j \neq y_i} \exp(\lambda \mathcal{S}'_{i,j})} \end{aligned} \quad (8)$$

Fig. 2 illustrates the differences between the SoftMax loss and the proposed losses.

Finally, we will show that the strategy of applying centers to construct triplet constraints can recover the constraints on original triplets.

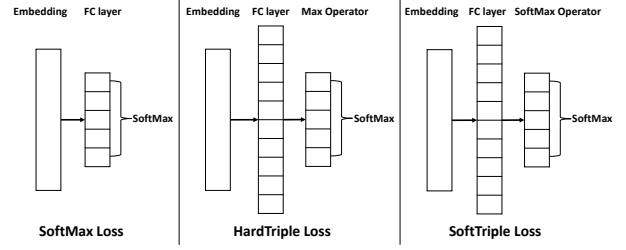


Figure 2. Illustration of differences between SoftMax loss and proposed losses. Compared with the SoftMax loss, we first increase the dimension of the FC layer to include multiple centers for each class (e.g., 2 centers per class in this example). Then, we obtain the similarity for each class by different operators. Finally, the distribution over different classes is computed with the similarity obtained from each class.

Theorem 1. Given two examples \mathbf{x}_i and \mathbf{x}_j that are from the same class and have the same nearest center and \mathbf{x}_k is from a different class, if the triple constant containing centers is satisfied

$$\mathbf{x}_i^\top \mathbf{w}_{y_i} - \mathbf{x}_i^\top \mathbf{w}_{y_k} \geq \delta$$

and we assume $\forall i, \|\mathbf{x}_i - \mathbf{w}_{y_i}\|_2 \leq \epsilon$, then we have

$$\mathbf{x}_i^\top \mathbf{x}_j - \mathbf{x}_i^\top \mathbf{x}_k \geq \delta - 2\epsilon$$

Proof.

$$\begin{aligned} \mathbf{x}_i^\top \mathbf{x}_j - \mathbf{x}_i^\top \mathbf{x}_k &= \mathbf{x}_i^\top (\mathbf{x}_j - \mathbf{w}_{y_i}) + \mathbf{x}_i^\top \mathbf{w}_{y_i} - \mathbf{x}_i^\top \mathbf{x}_k \\ &\geq \mathbf{x}_i^\top (\mathbf{x}_j - \mathbf{w}_{y_i}) + \mathbf{x}_i^\top (\mathbf{w}_{y_k} - \mathbf{x}_k) + \delta \\ &\geq \delta - \|\mathbf{x}_i\|_2 \|\mathbf{x}_j - \mathbf{w}_{y_i}\|_2 - \|\mathbf{x}_i\|_2 \|\mathbf{w}_{y_k} - \mathbf{x}_k\|_2 \\ &= \delta - \|\mathbf{x}_j - \mathbf{w}_{y_i}\|_2 - \|\mathbf{w}_{y_k} - \mathbf{x}_k\|_2 \geq \delta - 2\epsilon \end{aligned}$$

□

Theorem 1 demonstrates that optimizing the triplets consisting of centers with a margin δ can reserve the large margin property on the original triplet constraints. It also implies that more centers can be helpful to reduce the intra-class variance ϵ . In the extreme case that the number of centers is equal to the number of examples, ϵ becomes zero. However, adding more centers will increase the size of the last fully connected layer and make the optimization slow and computation expensive. Besides, it may incur the over-fitting problem.

Therefore, we have to choose an appropriate number of centers for each class that can have a small approximation error while keeping a compact set of centers. We will demonstrate the strategy in the next subsection.

3.2. Adaptive Number of Centers

Finding an appropriate number of centers for data is a challenging problem that also appears in unsupervised

learning, e.g., clustering. The number of centers K trades between the efficiency and effectiveness. In conventional DML algorithms, K equals to the number of original examples. It makes the number of total triplet constraints up to cubic of the number of original examples. In SoftMax loss, $K = 1$ reduces the number of constraints to be linear in the number of original examples, which is efficient but can be ineffective. Without the prior knowledge about the distribution of each class, it is hard to set K precisely.

Different from the strategy of setting the appropriate K for each class, we propose to set a sufficiently large K and then encourage similar centers to merge with each other. It can keep the diversity in the generated centers while shrinking the number of unique centers.

For each center \mathbf{w}_j^t , we can generate a matrix as

$$M_j^t = [\mathbf{w}_j^1 - \mathbf{w}_j^t, \dots, \mathbf{w}_j^K - \mathbf{w}_j^t]^\top$$

If \mathbf{w}_j^s and \mathbf{w}_j^t are similar, they can be collapsed to be the same one such that $\|\mathbf{w}_j^s - \mathbf{w}_j^t\|_2 = 0$, which is the L_2 norm of the s -th row in the matrix M_j^t . Therefore, we regularize the L_2 norm of rows in M_j^t to obtain a sparse set of centers, which can be written as the $L_{2,1}$ norm

$$\|M_j^t\|_{2,1} = \sum_s^K \|\mathbf{w}_j^s - \mathbf{w}_j^t\|_2$$

By accumulating $L_{2,1}$ norm over multiple centers, we can have the regularizer for the j -th class as

$$R(\mathbf{w}_j^1, \dots, \mathbf{w}_j^K) = \sum_t^K \|M_j^t\|_{2,1}$$

Since \mathbf{w} has the unit length, the regularizer is simplified as

$$R(\mathbf{w}_j^1, \dots, \mathbf{w}_j^K) = \sum_{t=1}^K \sum_{s=t+1}^K \sqrt{2 - 2\mathbf{w}_j^{s\top} \mathbf{w}_j^t} \quad (9)$$

With the regularizer, our final objective becomes

$$\min \frac{1}{N} \sum_i \ell_{\text{SoftTriple}}(\mathbf{x}_i) + \frac{\tau \sum_j^C R(\mathbf{w}_j^1, \dots, \mathbf{w}_j^K)}{CK(K-1)} \quad (10)$$

where N is the number of total examples.

4. Experiments

We conduct experiments on three benchmark fine-grained visual categorization data sets: *CUB-2011*, *Cars196* and *SOP*. We follow the settings in other works [3, 14] for the fair comparison. Specifically, we adopt the Inception [25] with the batch normalization [5] as the backbone architecture. The parameters of the backbone are initialized with the model trained on the ImageNet ILSVRC

2012 data set [20] and then fine-tuned on the target data sets. The images are cropped to 224×224 as the input of the network. During training, only random horizontal mirroring and random crop are used as the data augmentation. A single center crop is taken for test. The model is optimized by Adam with the batch size as 32 and the number of epochs as 50. The initial learning rates for the backbone and centers are set to be $1e-4$ and $1e-2$, respectively. Then, they are divided by 10 at $\{20, 40\}$ epochs. Considering that images in *CUB-2011* and *Cars196* are similar to those in ImageNet, we freeze BN on these two data sets and keep BN training on the rest one. Embeddings of examples and centers have the unit length in the experiments.

We compare the proposed triplet loss to the normalized SoftMax loss. The SoftMax loss in Eqn. 3 is denoted as **SoftMax_{norm}**. We refer the objective in Eqn. 10 as **Soft-Triple**. We set $\tau = 0.2$ and $\gamma = 0.1$ for SoftTriple. Besides, we set a small margin as $\delta = 0.01$ to break the tie explicitly. The number of centers is set to $K = 10$.

We evaluate the performance of the learned embeddings from different methods on the tasks of retrieval and clustering. For retrieval task, we use the Recall@ k metric as in [24]. The quality of clustering is measured by the Normalized Mutual Information (NMI) [13]. Given the clustering assignment $\mathbb{C} = \{c_1, \dots, c_n\}$ and the ground-truth label $\Omega = \{y_1, \dots, y_n\}$, NMI is computed as $\text{NMI} = \frac{2I(\Omega; \mathbb{C})}{H(\Omega) + H(\mathbb{C})}$, where $I(\cdot, \cdot)$ measures the mutual information and $H(\cdot)$ denotes the entropy.

4.1. CUB-2011

First, we compare the methods on a fine-grained birds data set *CUB-2011* [26]. It consists of 200 species of birds and 11,788 images. Following the common practice, we split the data set as that the first 100 classes are used for training and the rest are used for test. We note that different works report the results with different dimension of embeddings while the size of embeddings has a significant impact on the performance. For fair comparison, we report the results for the dimension of 64, which is adopted by many existing methods and the results with 512 feature embeddings, which reports the state-of-the-art results on most of data sets.

Table 1 summarizes the results with 64 embeddings. Note that Npairs* applies the multi-scale test while all other methods take a single crop test. For SemiHard [21], we report the result recorded in [23]. First, it is surprising to observe that the performance of SoftMax_{norm} surpasses that of the existing metric learning methods. It is potentially due to the fact that SoftMax loss optimizes the relations of examples as a smoothed triplet loss, which is analyzed in Proposition 1. Second, SoftTriple demonstrates the best performance among all benchmark methods. Compared to ProxyNCA, SoftTriple improves the state-of-the-

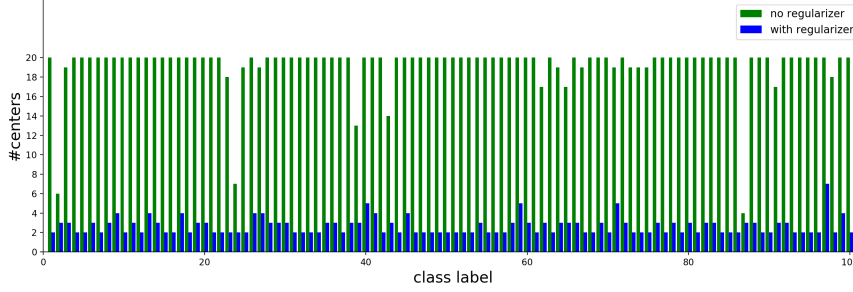


Figure 3. Comparison of the number of unique centers in each class on *CUB-2011*. The initial number of centers is set to 20.

art performance by 10% on R@1. Besides, it is 2% better than $\text{SoftMax}_{\text{norm}}$. It verifies that SoftMax loss cannot capture the complex geometry of real-world data set with a single center for each class. When increasing the number of centers, SoftTriple can depict the inherent structure of data better. Finally, both of SoftMax and SoftTriple show the superior performance compared to existing methods. It demonstrates that meaningful embeddings can be learned without a sampling phase.

Table 1. Comparison on *CUB-2011*. The dimension of the embeddings for all methods is 64.

Methods	R@1	R@2	R@4	R@8	NMI
SemiHard [21]	42.6	55.0	66.4	77.2	55.4
LiftedStruct [24]	43.6	56.6	68.6	79.6	56.5
Clustering [23]	48.2	61.4	71.8	81.9	59.2
Npairs* [22]	51.0	63.3	74.3	83.2	60.4
ProxyNCA [14]	49.2	61.9	67.9	72.4	59.5
$\text{SoftMax}_{\text{norm}}$	57.8	70.0	80.1	87.9	65.3
SoftTriple	60.1	71.9	81.2	88.5	66.2

Table 2 compares SoftTriple with 512 embeddings to the methods with large embeddings. HDC [32] applies the dimension as 384. Margin [12] takes 128 dimension of embeddings and uses ResNet50 [4] as the backbone. HTL [3] sets the dimension of embeddings to 512 and reports the state-of-the-art result on the backbone of Inception. With the large number of embeddings, it is obvious that all methods outperform existing DML methods with 64 embeddings in Table 1. It is as expected since the high dimensional space can separate examples better, which is consistent with the observation in other work [24]. Compared with other methods, the R@1 of SoftTriple improves more than 8% over HTL that has the same backbone as SoftTriple. It also increases R@1 by about 2% over Margin, which applies a stronger backbone than Inception. It shows that SoftTriple loss is applicable with large embeddings.

To validate the effect of the proposed regularizer, we compare the number of unique centers for each class in Fig. 3. We set a larger number of centers as $K = 20$ to make the results explicit and then run SoftTriple with and without the regularizer in Eqn. 9. Fig. 3 illustrates that the

Table 2. Comparison on *CUB-2011* with large embeddings. “-” means the result is not available.

Methods	R@1	R@2	R@4	R@8	NMI
HDC [32]	53.6	65.7	77.0	85.6	-
Margin [12]	63.6	74.4	83.1	90.0	69.0
HTL [3]	57.1	68.8	78.7	86.5	-
$\text{SoftMax}_{\text{norm}}$	64.2	75.6	84.3	90.2	68.3
SoftTriple	65.4	76.4	84.5	90.4	69.3

one without regularizer will hold a set of similar centers. In contrast, SoftTriple with the regularizer can shrink the size of centers significantly and make the optimization effective.

Besides, we demonstrate the R@1 of SoftTriple with varying the number of centers in Fig. 4. Red line denotes SoftTriple loss equipped with the regularizer while blue dashed line has no regularizer. We find that when increasing the number of centers from 1 to 10, the performance of SoftTriple is improved significantly, which confirms that with leveraging multiple centers, the learned embeddings can capture the data distribution better. If adding more centers, the performance of SoftTriple almost remains the same and it shows that the regularizer can help to learn the compact set of centers and will not be influenced by the initial number of centers. On the contrary, without the regularizer, the blue dashed line illustrates that the performance will degrade due to overfitting when the number of centers are over-parameterized.

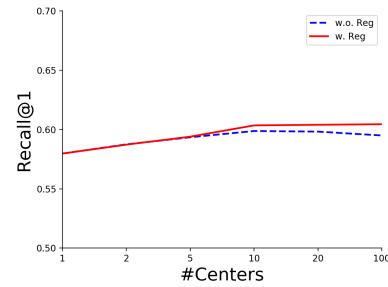


Figure 4. Illustration of SoftTriple with different number of centers and the influence of the regularizer. With the proposed regularizer as denoted by the red line, the performance is stable to the initial number of centers K when it is sufficiently large.

Finally, we illustrate the examples of retrieved images in Fig. 5. The first column indicates the query image. The columns 2-4 show the most similar images retrieved according to the embeddings learned by SoftMax_{norm}. The last four columns are the similar images returned by using the embeddings from SoftTriple. Evidently, embeddings from SoftMax_{norm} can obtain the meaningful neighbors while the objective is for classification. Besides, SoftTriple improves the performance and can eliminate the images from different classes among the top of retrieved images, which are highlighted with red bounding boxes in SoftMax_{norm}.



Figure 5. Examples of retrieved most similar images with the learned embeddings from SoftMax_{norm} and SoftTriple. The images from the classes that are different from the query image are highlighted by red bounding boxes.

4.2. Cars196

Then, we conduct the experiments on *Cars196* data set [6], which contains 196 models of cars and 16,185 images. We use the first 98 classes for training and the rest for test. Table 3 summarizes the performance with 64 embeddings. The observation is similar as for *CUB-2011*. SoftMax_{norm} shows the superior performance and is 3% better than ProxyNCA on R@1. Additionally, SoftTriple can further improve the performance by about 2%, which demonstrates the effectiveness of the proposed loss function.

Table 3. Comparison on *Cars196*. The dimension is 64.

Methods	R@1	R@2	R@4	R@8	NMI
SemiHard [21]	51.5	63.8	73.5	82.4	53.4
LiftedStruct [24]	53.0	65.7	76.0	84.3	56.9
Clustering [23]	58.1	70.6	80.3	87.8	59.0
Npairs* [22]	71.1	79.7	86.5	91.6	64.0
ProxyNCA [14]	73.2	82.4	86.4	88.7	64.9
SoftMax _{norm}	76.8	85.6	91.3	95.2	66.7
SoftTriple	78.6	86.6	91.8	95.4	67.0

In Table 4, we present the comparison with large dimension of embeddings. The number of embeddings for all

methods in the comparison is the same as described in the experiments on *CUB-2011*. On this data set, HTL [3] reports the state-of-the-art result while SoftTriple outperforms it and increases R@1 by 3%.

Table 4. Comparison on *Cars196* with large embeddings.

Methods	R@1	R@2	R@4	R@8	NMI
HDC [32]	73.7	83.2	89.5	93.8	-
Margin [12]	79.6	86.5	91.9	95.1	69.1
HTL [3]	81.4	88.0	92.7	95.7	-
SoftMax _{norm}	83.2	89.5	94.0	96.6	69.7
SoftTriple	84.5	90.7	94.5	96.9	70.1

4.3. Stanford Online Products

Finally, we evaluate the performance of different methods on the Stanford Online Products (*SOP*) data set [24]. It contains 120,053 product images downloaded from eBay.com and includes 22,634 classes. We adopt the standard splitting, where 11,318 classes are used for training and the rest for test. Note that each class has about 5 images, so we set $K = 2$ for this data set and discard the regularizer. We also increase the initial learning rate for centers from 0.01 to 0.1.

We first report the results with 64 embeddings in Table 5. In this comparison, SoftMax_{norm} is 2% better than ProxyNCA on R@1. By simply increasing the number of centers from 1 to 2, we observe that SoftTriple gains another 0.4% on R@1. It confirms that multiple centers can help to capture the data structure better.

Table 6 states the performance with large embeddings. We can get a similar conclusion as in Table 5. Both SoftMax_{norm} and SoftTriple outperform the state-of-the-art methods. SoftTriple improves the state-of-the-art by more than 3% on R@1. It demonstrates the advantage of learning embeddings without sampling triplet constraints.

Table 5. Comparison on *SOP*. The dimension is 64.

Methods	R@1	R@10	R@100	NMI
SemiHard [21]	66.7	82.4	91.9	89.5
LiftedStruct [24]	62.5	80.8	91.9	88.7
Clustering [23]	67.0	83.7	93.2	89.5
ProxyNCA [14]	73.7	-	-	90.6
SoftMax _{norm}	75.9	88.8	95.2	91.5
SoftTriple	76.3	89.1	95.3	91.7

Table 6. Comparison on *SOP* with large embeddings.

Methods	R@1	R@10	R@100	NMI
Npairs* [22]	67.7	83.8	93.0	88.1
HDC [32]	69.5	84.4	92.8	-
Margin [12]	72.7	86.2	93.8	90.7
HTL [3]	74.8	88.3	94.8	-
SoftMax _{norm}	78.0	90.2	96.0	91.9
SoftTriple	78.3	90.3	95.9	92.0

5. Conclusion

Sampling triplets from a mini-batch of data can degrade the performance of deep metric learning due to its poor coverage over the whole data set. To address the problem, we propose the novel SoftTriple loss to learn the embeddings without sampling. By representing each class with multiple centers, the loss can be optimized with triplets defined with the similarities between the original examples and classes. Since centers are encoded in the last fully connected layer, we can learn embeddings with the standard SGD training pipeline for classification and eliminate the sampling phase. The consistent improvement from SoftTriple over fine-grained benchmark data sets confirms the effectiveness of the proposed loss function. Since SoftMax loss is prevalently applied for classification, SoftTriple loss can also be applicable for that. Evaluating SoftTriple on the classification task can be our future work.

References

- [1] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004. 4
- [2] Jiankang Deng, Jia Guo, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. *CoRR*, abs/1801.07698, 2018. 2
- [3] Weifeng Ge, Weilin Huang, Dengke Dong, and Matthew R. Scott. Deep metric learning with hierarchical triplet loss. In *ECCV*, pages 272–288, 2018. 3, 6, 7, 8
- [4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 7
- [5] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015. 6
- [6] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013. 8
- [7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012. 2, 3
- [8] Daryl Lim, Gert R. G. Lanckriet, and Brian McFee. Robust structural metric learning. In *ICML*, pages 615–623, 2013. 2
- [9] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*, pages 6738–6746, 2017. 2
- [10] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In *ICML*, pages 507–516, 2016. 2
- [11] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. National Institute of Science of India, 1936. 1
- [12] R. Manmatha, Chao-Yuan Wu, Alexander J. Smola, and Philipp Krähenbühl. Sampling matters in deep embedding learning. In *ICCV*, pages 2859–2867, 2017. 2, 3, 7, 8
- [13] Christopher Manning, Prabhakar Raghavan, and Hinrich Schütze. Introduction to information retrieval. *Natural Language Engineering*, 16(1):100–103, 2010. 6
- [14] Yair Movshovitz-Attias, Alexander Toshev, Thomas K. Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. In *ICCV*, pages 360–368, 2017. 2, 3, 4, 6, 7, 8
- [15] Omkar M. Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. In *BMVC*, pages 41.1–41.12, 2015. 1, 2, 3
- [16] Qi Qian, Rong Jin, Jinfeng Yi, Lijun Zhang, and Shenghuo Zhu. Efficient distance metric learning by adaptive sampling and mini-batch stochastic gradient descent (SGD). *Machine Learning*, 99(3):353–372, 2015. 1, 2
- [17] Qi Qian, Rong Jin, Shenghuo Zhu, and Yuanqing Lin. Fine-grained visual categorization via multi-stage metric learning. In *CVPR*, pages 3716–3724, 2015. 1, 2
- [18] Qi Qian, Jiasheng Tang, Hao Li, Shenghuo Zhu, and Rong Jin. Large-scale distance metric learning with uncertainty. In *CVPR*, pages 8542–8550, 2018. 2, 3
- [19] Oren Rippel, Manohar Paluri, Piotr Dollár, and Lubomir D. Bourdev. Metric learning with adaptive density discrimination. *ICLR*, 2016. 2, 3
- [20] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 6
- [21] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823, 2015. 1, 2, 3, 6, 7, 8
- [22] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *NIPS*, pages 1849–1857, 2016. 2, 7, 8
- [23] Hyun Oh Song, Stefanie Jegelka, Vivek Rathod, and Kevin Murphy. Deep metric learning via facility location. In *CVPR*, pages 2206–2214, 2017. 6, 7, 8
- [24] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *CVPR*, pages 4004–4012, 2016. 1, 2, 3, 6, 7, 8
- [25] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, pages 1–9, 2015. 6
- [26] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, 2011. 6
- [27] Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. Additive margin softmax for face verification. *IEEE Signal Process. Lett.*, 25(7):926–930, 2018. 2
- [28] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*, pages 5265–5274, 2018. 2

- [29] Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, 2009. [1](#), [2](#)
- [30] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, pages 499–515, 2016. [2](#)
- [31] Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart J. Russell. Distance metric learning with application to clustering with side-information. In *NIPS*, pages 505–512, 2002. [1](#), [2](#)
- [32] Yuhui Yuan, Kuiyuan Yang, and Chao Zhang. Hard-aware deeply cascaded embedding. In *ICCV*, pages 814–823, 2017. [7](#), [8](#)