Deep Metric Learning with Hierarchical Triplet Loss

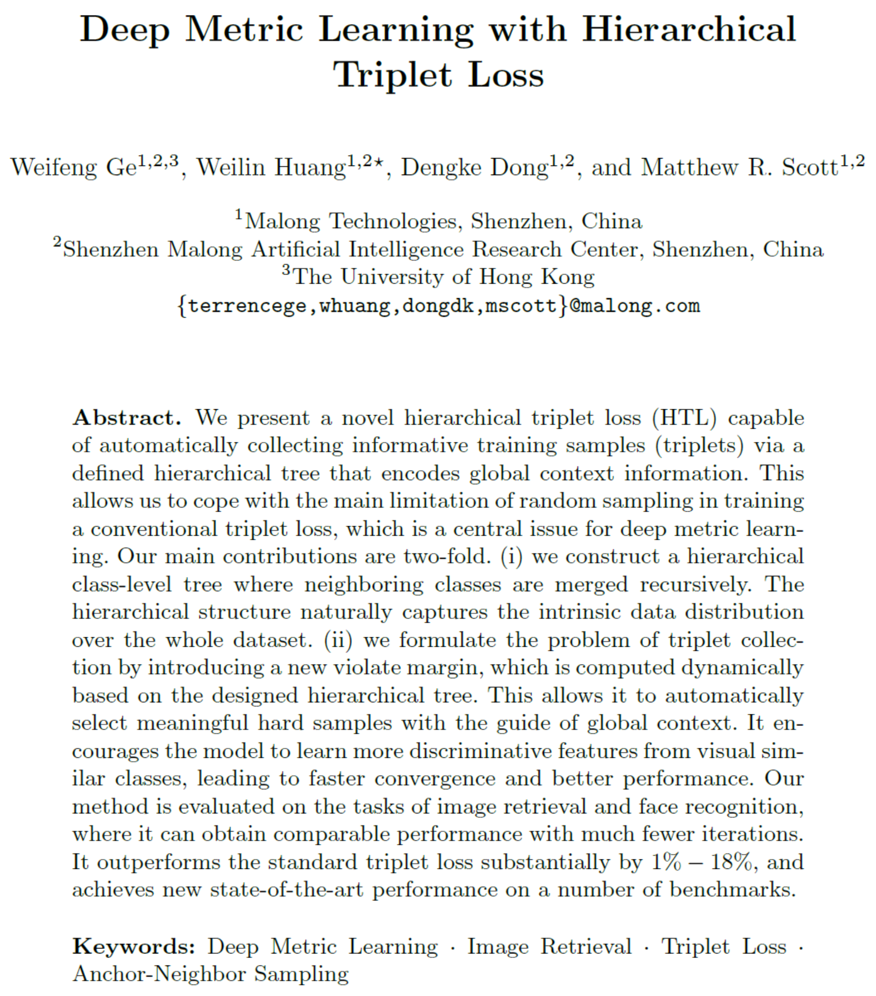
Sanghyun Seo

Jan 18, 2018

Dongguk University

Artificial Intelligence Laboratory

- Reference
  - Ge, Weifeng, et al. "Deep metric learning with hierarchical triplet loss." *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.
- Keywords
  - Deep Metric Learning
  - Image Retrieval
  - Triplet Loss
  - Anchor-Neighbor Sampling



Deep Metric Learning with Hierarchical Triplet Loss

Weifeng Ge[1,2,3], Weilin Huang[1,2⋆], Dengke Dong[1,2], and Matthew R. Scott[1,2]

[1]Malong Technologies, Shenzhen, China
[2]Shenzhen Malong Artificial Intelligence Research Center, Shenzhen, China
[3]The University of Hong Kong
{terrencege,whuang,dongdk,mscott}@malong.com

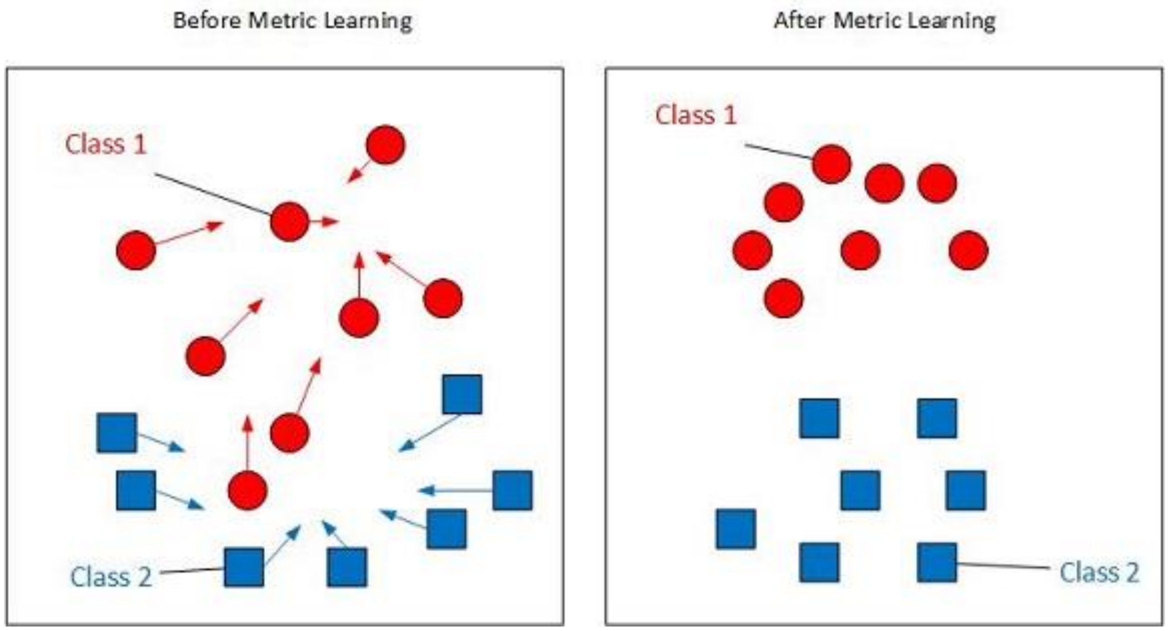
Abstract. We present a novel hierarchical triplet loss (HTL) capable of automatically collecting informative training samples (triplets) via a defined hierarchical tree that encodes global context information. This allows us to cope with the main limitation of random sampling in training a conventional triplet loss, which is a central issue for deep metric learning. Our main contributions are two-fold. (i) we construct a hierarchical class-level tree where neighboring classes are merged recursively. The hierarchical structure naturally captures the intrinsic data distribution over the whole dataset. (ii) we formulate the problem of triplet collection by introducing a new violate margin, which is computed dynamically based on the designed hierarchical tree. This allows it to automatically select meaningful hard samples with the guide of global context. It encourages the model to learn more discriminative features from visual similar classes, leading to faster convergence and better performance. Our method is evaluated on the tasks of image retrieval and face recognition, where it can obtain comparable performance with much fewer iterations. It outperforms the standard triplet loss substantially by $1\% - 18\%$, and achieves new state-of-the-art performance on a number of benchmarks.

Keywords: Deep Metric Learning · Image Retrieval · Triplet Loss · Anchor-Neighbor Sampling

Contents

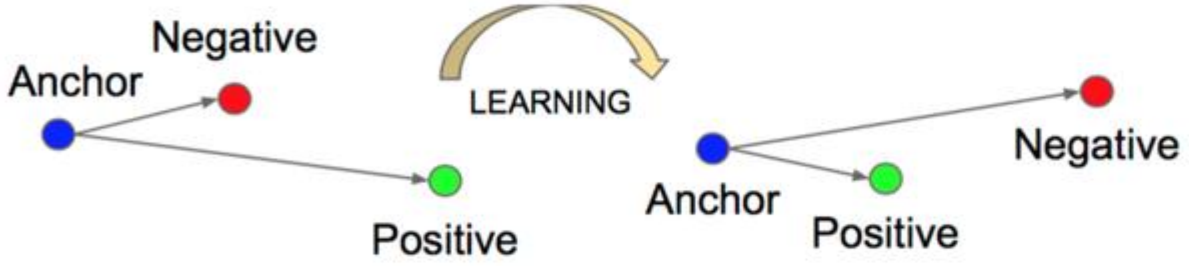
Introduction

- Distance metric learning
  - Given a similarity function, images with **similar content are projected onto neighboring locations on a manifold**, and images with **different semantic context are mapped apart from each other**.
  - Using **embedding model** such as deep neural networks



Introduction

- Loss function
  - Recently, there is a number of widely-used loss functions developed for deep metric learning, such as contrastive loss, **triplet loss** and quadruplet loss. These loss functions are calculated on correlated samples, with a common goal of encouraging samples from the same class to be closer, and pushing samples of different classes apart from each other, in a projected feature space.



$$Loss = \sum_{i=1}^{N} \left[\|f_i^a - f_i^p\|_2^2 - \|f_i^a - f_i^n\|_2^2 + \alpha \right]_+$$

Introduction

- Contribution of this paper
  - We propose a novel hierarchical triplet loss that allows the model to collect informative training samples with the guide of a **global class-level hierarchical tree**.
  - We formulate the problem of triplet collection **by introducing a new violate margin**, which is computed dynamically over the constructed hierarchical tree.
  - The proposed HTL is easily implemented, and **can be readily integrated into the standard triplet loss or other deep metric learning approaches**, such as contrastive loss, quadruplet loss, recent HDC and BIER.

Related Work

- Deep Metric Learning
  - Deep metric learning maps an image into a feature vector in a manifold space via deep neural networks. In this manifold space, the Euclidean distance (or the cosine distance) can be directly used as the distance metric between two points. The contribution of many deep metric learning algorithms, such as [26, 22, 5, 2, 3], is the design of a loss function that can learn more discriminant features. Since neural networks are usually trained using the stochastic gradient descent (SGD) in mini-batches, these loss functions are difficult to approximate the target of metric learning - pull samples with the same label into nearby points and push samples with different labels apart.

Related Work

- Informative Sample Selection
  - Given N training images, there are about O(N^2) pairs, O(N^3) triplets, and O(N^4) quadruplets → **It is infeasible to traverses all these training tuples during training.**
  - While it is rather inconvenient to take thousands of images in a mini-batch with a large-scale network, due to the **limitation of GPU memory**.
  - For deep metric learning**, it is of great importance to selecting informative training tuples.**

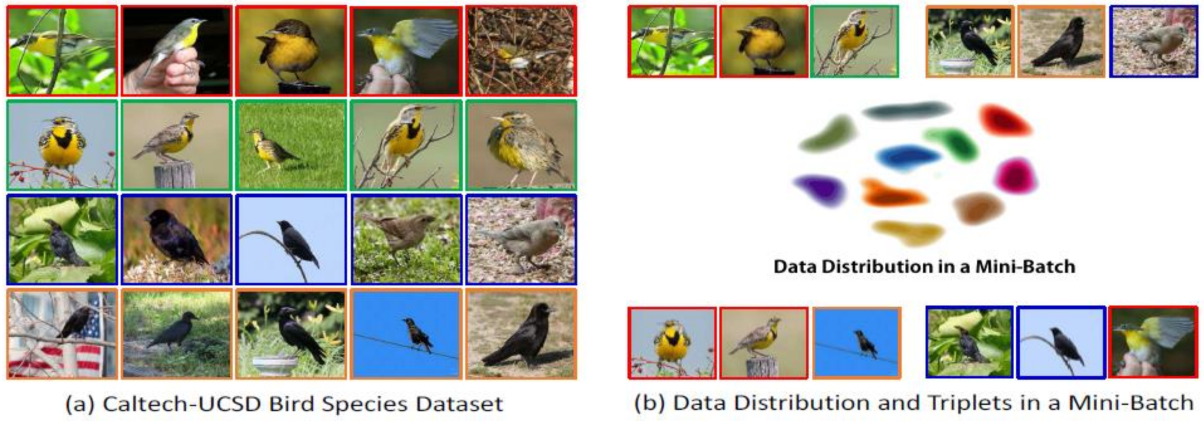
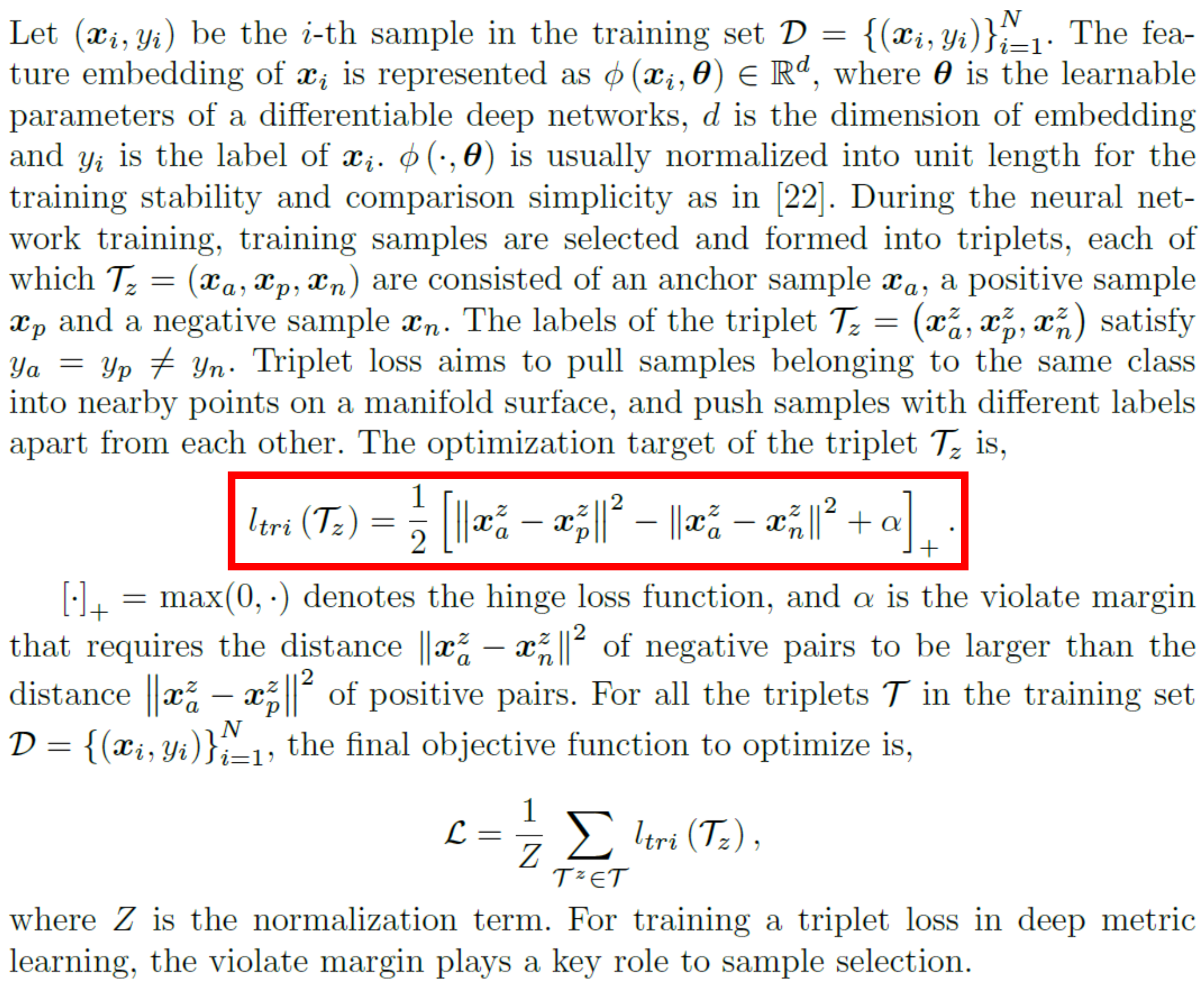

Data Distribution in a Mini-Batch
(a) Caltech-UCSD Bird Species Dataset
(b) Data Distribution and Triplets in a Mini-Batch

Fig. 1. (a) Caltech-UCSD Bird Species Dataset [31]. Images in each row are from the same class. There are four classes in different colors — red, green, blue and yellow. (b) Data distribution and triplets in a mini-batch. Triplets in the top row violate the triplet constrain in the traditional triplet loss. Triplets in the bottom row are ignored in the triplet loss, but are revisited in the hierarchical triplet loss.

Motivation: Challenges in Triplet Loss

- Standard Triplet Loss



Let $(\boldsymbol{x}_i, y_i)$ be the i-th sample in the training set $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$. The feature embedding of $\boldsymbol{x}_i$ is represented as $\phi(\boldsymbol{x}_i, \boldsymbol{\theta}) \in \mathbb{R}^d$, where $\boldsymbol{\theta}$ is the learnable parameters of a differentiable deep networks, d is the dimension of embedding and y_i is the label of $\boldsymbol{x}_i$. $\phi(\cdot, \boldsymbol{\theta})$ is usually normalized into unit length for the training stability and comparison simplicity as in [22]. During the neural network training, training samples are selected and formed into triplets, each of which $\mathcal{T}_z = (\boldsymbol{x}_a, \boldsymbol{x}_p, \boldsymbol{x}_n)$ are consisted of an anchor sample $\boldsymbol{x}_a$, a positive sample $\boldsymbol{x}_p$ and a negative sample $\boldsymbol{x}_n$. The labels of the triplet $\mathcal{T}_z = (\boldsymbol{x}_a^z, \boldsymbol{x}_p^z, \boldsymbol{x}_n^z)$ satisfy $y_a = y_p \neq y_n$. Triplet loss aims to pull samples belonging to the same class into nearby points on a manifold surface, and push samples with different labels apart from each other. The optimization target of the triplet $\mathcal{T}_z$ is,

$$\boxed{l_{tri}(\mathcal{T}_z) = \frac{1}{2}\left[\left\|\boldsymbol{x}_a^z - \boldsymbol{x}_p^z\right\|^2 - \left\|\boldsymbol{x}_a^z - \boldsymbol{x}_n^z\right\|^2 + \alpha\right]_+ .}$$

$[\cdot]_+ = \max(0, \cdot)$ denotes the hinge loss function, and α is the violate margin that requires the distance $\left\|\boldsymbol{x}_a^z - \boldsymbol{x}_n^z\right\|^2$ of negative pairs to be larger than the distance $\left\|\boldsymbol{x}_a^z - \boldsymbol{x}_p^z\right\|^2$ of positive pairs. For all the triplets $\mathcal{T}$ in the training set $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$, the final objective function to optimize is,

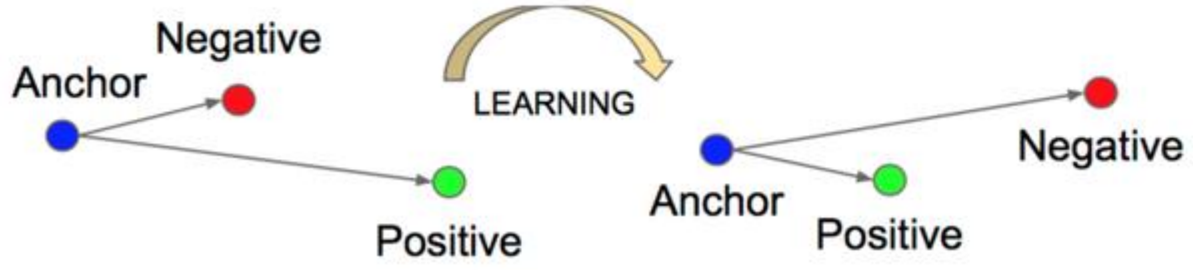
$$\mathcal{L} = \frac{1}{Z} \sum_{\mathcal{T}^z \in \mathcal{T}} l_{tri}(\mathcal{T}_z),$$

where Z is the normalization term. For training a triplet loss in deep metric learning, the violate margin plays a key role to sample selection.

Motivation: Challenges in Triplet Loss

- Challenge1: triplet loss with random sampling
  - For a training set D $= \{(x_i, y_i)\}_{k=1}^{N}$ with N samples, training a triplet, loss will generate $\mathrm{O}(N^3)$ triplets, which is infeasible to put all triplets into a single mini-batch.
  - When **we sample the triplets over the whole training set randomly**, it **has a risk of slow convergence and pool local optima**. We identify the problem that most of training samples obey the violate margin when the model starts to converge.
  - **These samples can not contribute gradients to the learning process**, and thus are less informative, but can dominate the training process, which significantly degrades the model capability, with a slow convergence.

Random sampling? → less informative



Motivation: Challenges in Triplet Loss

- Challenge2: risk of local optima
  - **In triplet loss, all triplet is treated equally**. As shown in Fig. 1, when the training goes after several epoches, **most of training triplets dose not contribute to the gradients** of learnable parameters in deep neural networks.

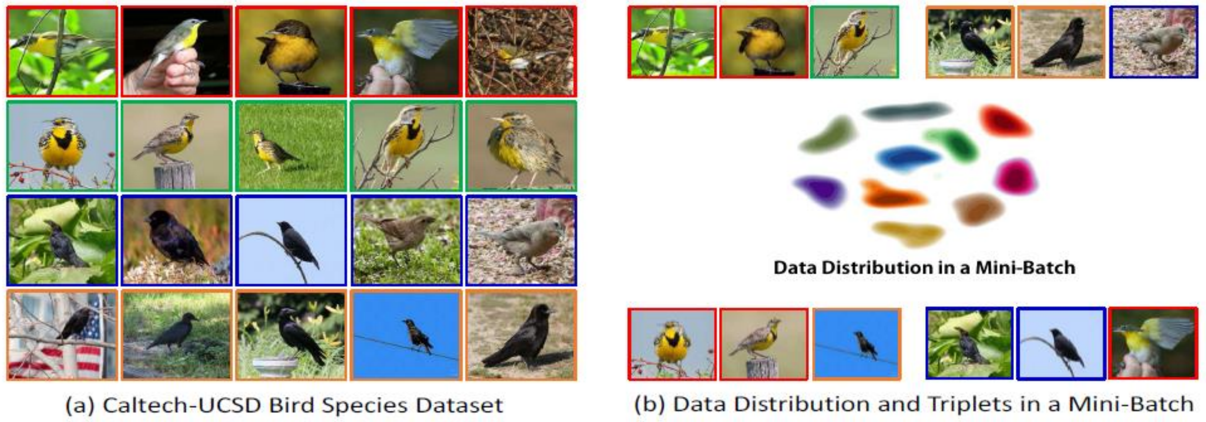
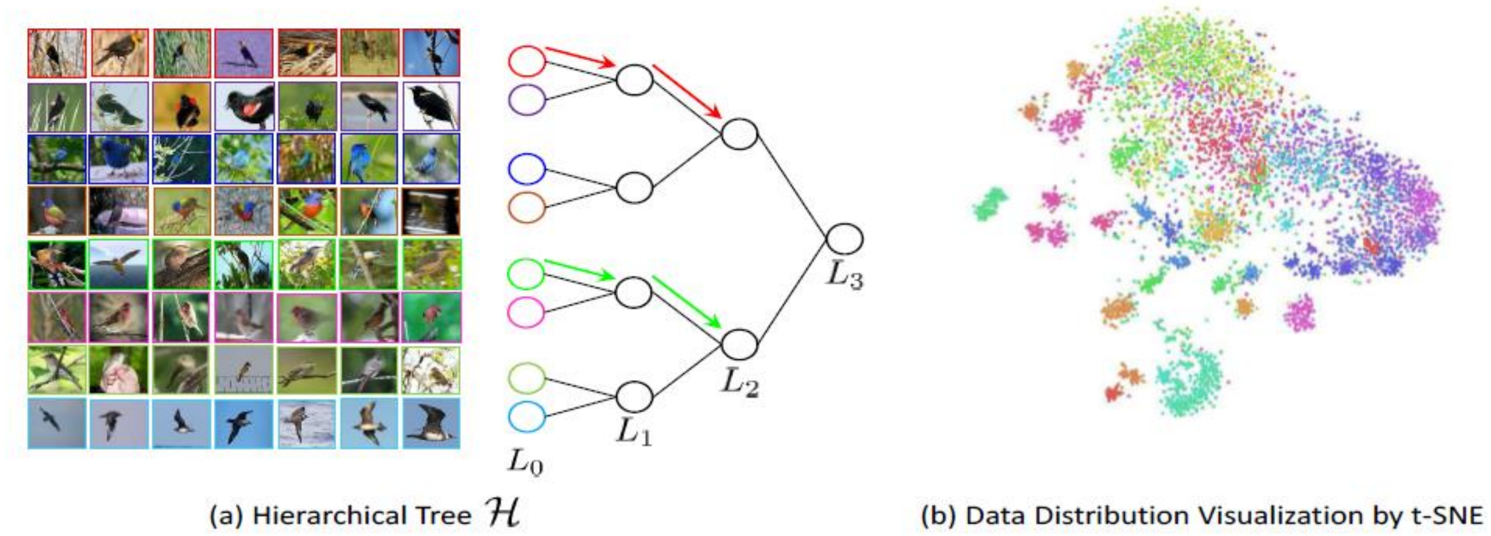

Data Distribution in a Mini-Batch
(a) Caltech-UCSD Bird Species Dataset
(b) Data Distribution and Triplets in a Mini-Batch

Fig. 1. (a) Caltech-UCSD Bird Species Dataset [31]. Images in each row are from the same class. There are four classes in different colors — red, green, blue and yellow. (b) Data distribution and triplets in a mini-batch. Triplets in the top row violate the triplet constrain in the traditional triplet loss. Triplets in the bottom row are ignored in the triplet loss, but are revisited in the hierarchical triplet loss.

Hierarchical Triplet Loss

- Manifold structure in Hierarchy
  - We describe details of the proposed hierarchical triplet loss, which contains two main components, <u>**constructing a hierarchical class tree**</u> and <u>**formulating the hierarchical triplet loss with a new violate margin.**</u>

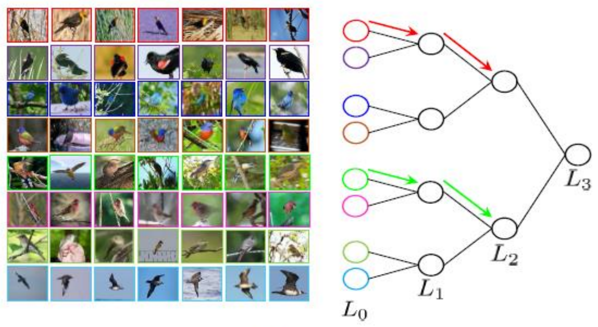


L_0
L_1
L_2
L_3
(a) Hierarchical Tree $\mathcal{H}$
(b) Data Distribution Visualization by t-SNE

Fig. 2. (a) A toy example of the hierarchical tree $\mathcal{H}$. Different colors represent different image classes in CUB-200-2011 [31]. The leaves are the image classes in the training set. Then they are merged recursively until to the root node. (b) The training data distribution of 100 classes visualized by using t-SNE [16] to reduce the dimension of triplet embedding from 512 to 2.

Hierarchical Triplet Loss

- Manifold structure in Hierarchy



(a) Hierarchical Tree $\mathcal{H}$

We construct a global hierarchy at the class level. Given a neural network $\phi_t(\cdot, \boldsymbol{\theta}) (\in \mathbb{R}^d)$ pre-trained using the traditional triplet loss, we get the hierarchical data structure based on sample rules. Denote the deep feature of a sample $\boldsymbol{x}_i$ as $\boldsymbol{r}_i = \phi_t(\boldsymbol{x}_i, \boldsymbol{\theta})$. We first calculate a distance matrix of $\mathcal{C}$ classes in the whole training set $\mathcal{D}$. The distance between the p-th class and the q-th class is computed as,

$$d(p, q) = \frac{1}{n_p n_q} \sum_{i \in p, j \in q} \|\boldsymbol{r}_i - \boldsymbol{r}_j\|^2,$$

where n_p and n_q are the numbers of training samples in the p-th and the q-th classes respectively. Since the deep feature $\boldsymbol{r}_i$ is normalized into unit length, the value of the interclass distance $d(p, q)$ varies from 0 to 4.

Hierarchical Triplet Loss

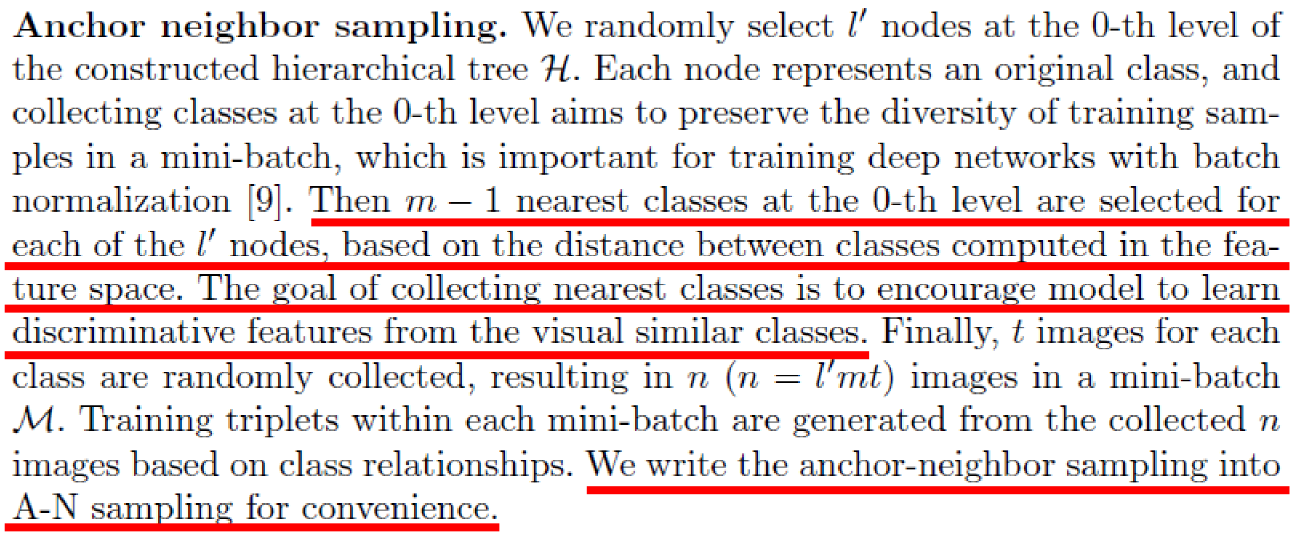
- Manifold structure in Hierarchy
  - They build hierarchical manifold structure by creating a **hierarchical tree**, according to the **computed interclass distances**. The leaves of the hierarchical tree are the original image classes, where each class represents a leave node at the 0-th level
  - **Then hierarchy is created by recursively merging** the leave notes at different levels, **based on the computed distance matrix**
  - The hierarchical tree is set into L levels, and the average inner distance d_0 is used as the threshold for merging the nodes at the 0-th level.

$$d_0 = \frac{1}{\mathcal{C}} \sum_{c=1}^{\mathcal{C}} \left(\frac{1}{n_c^2 - n_c} \sum_{i \in c, j \in c} \|\boldsymbol{r}_i - \boldsymbol{r}_j\|^2 \right).$$

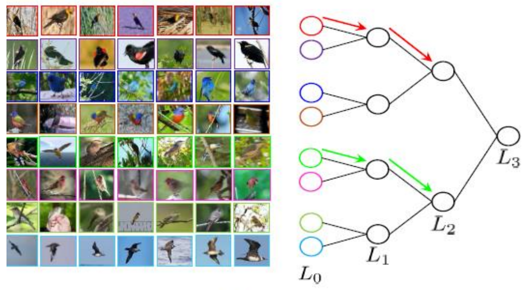
where n_c is the number of samples in the c-th class. Then the nodes are merged with different thresholds. At the l-th level of the hierarchical tree, the merging threshold is set to $d_l = \frac{l(4-d_0)}{L} + d_0$. Two classes with a distance less than d_l are merged into a node at the l-th level. The node number at the l-th level is N_l. The nodes are merged from the 0-th level to the L-th level. Finally, we generate a hierarchical tree $\mathcal{H}$ which starts from the leave nodes of original image classes to a final top node, as shown in Fig. 2 (a). The constructed hierarchical tree captures class relationships over the whole dataset, and it is updated interactively at the certain iterations over the training.

Hierarchical Triplet Loss

- Hierarchical Triplet Loss
  - They formulate the problem of triplet collection into a hierarchical triplet loss.
  - They introduce a **dynamical violate margin**, which is the main difference from the conventional triplet loss using a constant violate margin.



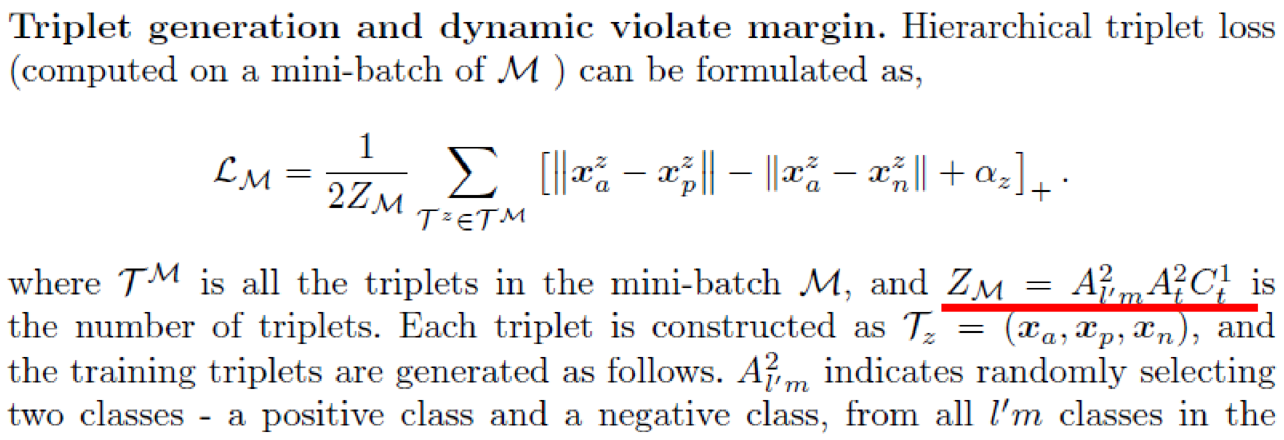
Anchor neighbor sampling. We randomly select l' nodes at the 0-th level of the constructed hierarchical tree $\mathcal{H}$. Each node represents an original class, and collecting classes at the 0-th level aims to preserve the diversity of training samples in a mini-batch, which is important for training deep networks with batch normalization [9]. Then $m-1$ nearest classes at the 0-th level are selected for each of the l' nodes, based on the distance between classes computed in the feature space. The goal of collecting nearest classes is to encourage model to learn discriminative features from the visual similar classes. Finally, t images for each class are randomly collected, resulting in n $(n = l'mt)$ images in a mini-batch $\mathcal{M}$. Training triplets within each mini-batch are generated from the collected n images based on class relationships. We write the anchor-neighbor sampling into A-N sampling for convenience.



(a) Hierarchical Tree $\mathcal{H}$

Hierarchical Triplet Loss

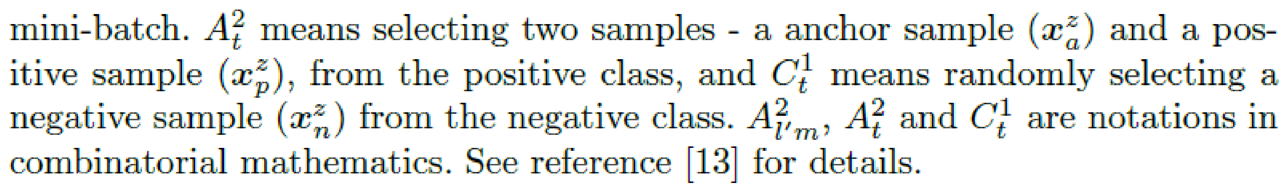
- Hierarchical Triplet Loss
  - Dynamic violate margin



Triplet generation and dynamic violate margin. Hierarchical triplet loss (computed on a mini-batch of $\mathcal{M}$) can be formulated as,

$$\mathcal{L}_{\mathcal{M}} = \frac{1}{2Z_{\mathcal{M}}} \sum_{\mathcal{T}^z \in \mathcal{T}^{\mathcal{M}}} \left[\|\boldsymbol{x}_a^z - \boldsymbol{x}_p^z\| - \|\boldsymbol{x}_a^z - \boldsymbol{x}_n^z\| + \alpha_z \right]_+ .$$

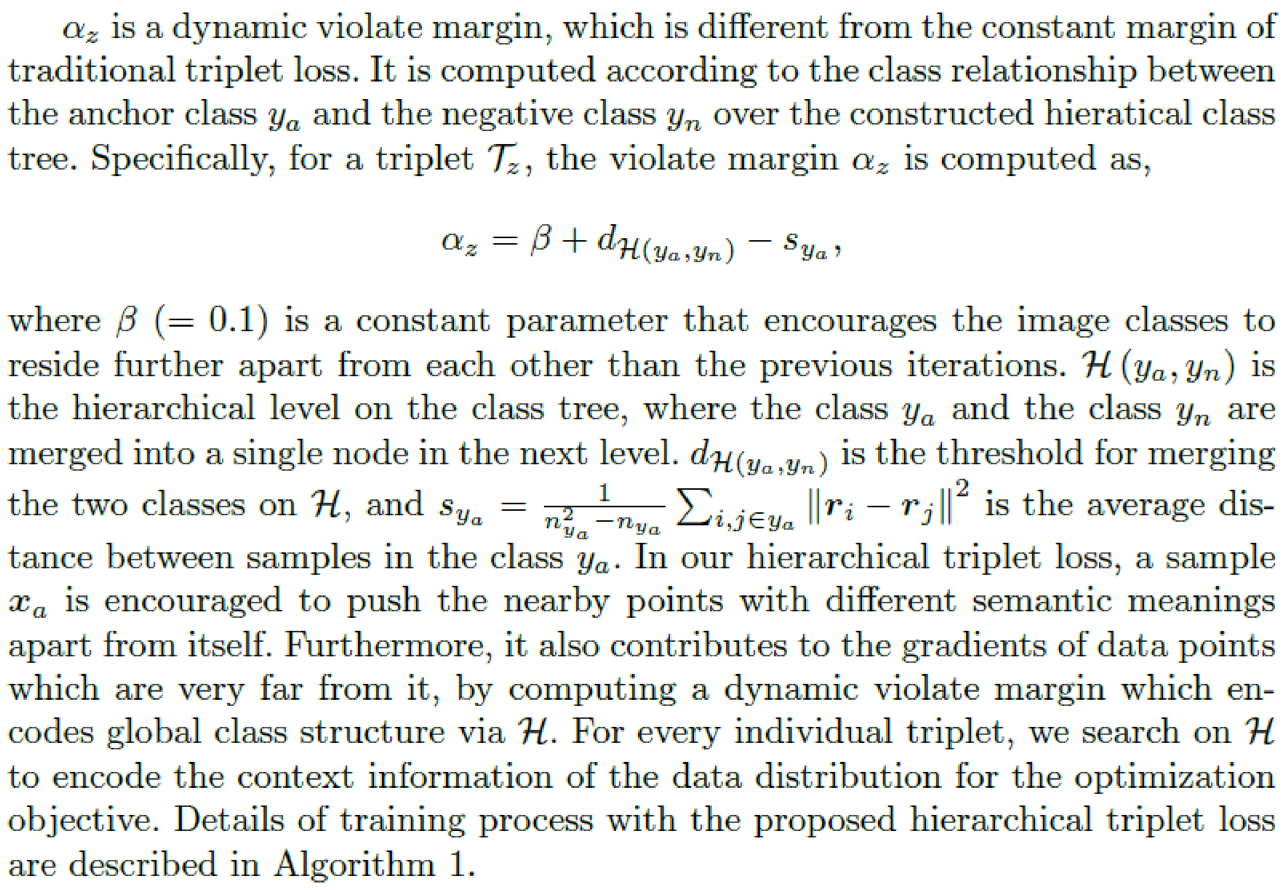
where $\mathcal{T}^{\mathcal{M}}$ is all the triplets in the mini-batch $\mathcal{M}$, and $\underline{Z_{\mathcal{M}} = A^2_{l'm} A^2_t C^1_t}$ is the number of triplets. Each triplet is constructed as $\mathcal{T}_z = (\boldsymbol{x}_a, \boldsymbol{x}_p, \boldsymbol{x}_n)$, and the training triplets are generated as follows. $A^2_{l'm}$ indicates randomly selecting two classes - a positive class and a negative class, from all $l'm$ classes in the



mini-batch. A^2_t means selecting two samples - a anchor sample ($\boldsymbol{x}_a^z$) and a positive sample ($\boldsymbol{x}_p^z$), from the positive class, and C^1_t means randomly selecting a negative sample ($\boldsymbol{x}_n^z$) from the negative class. $A^2_{l'm}$, A^2_t and C^1_t are notations in combinatorial mathematics. See reference [13] for details.

Hierarchical Triplet Loss

- Hierarchical Triplet Loss



α_z is a dynamic violate margin, which is different from the constant margin of traditional triplet loss. It is computed according to the class relationship between the anchor class y_a and the negative class y_n over the constructed hieratical class tree. Specifically, for a triplet $\mathcal{T}_z$, the violate margin α_z is computed as,

$$\alpha_z = \beta + d_{\mathcal{H}(y_a, y_n)} - s_{y_a},$$

where β $(= 0.1)$ is a constant parameter that encourages the image classes to reside further apart from each other than the previous iterations. $\mathcal{H}(y_a, y_n)$ is the hierarchical level on the class tree, where the class y_a and the class y_n are merged into a single node in the next level. $d_{\mathcal{H}(y_a, y_n)}$ is the threshold for merging the two classes on $\mathcal{H}$, and $s_{y_a} = \frac{1}{n_{y_a}^2 - n_{y_a}} \sum_{i,j \in y_a} \|\boldsymbol{r}_i - \boldsymbol{r}_j\|^2$ is the average distance between samples in the class y_a. In our hierarchical triplet loss, a sample x_a is encouraged to push the nearby points with different semantic meanings apart from itself. Furthermore, it also contributes to the gradients of data points which are very far from it, by computing a dynamic violate margin which encodes global class structure via $\mathcal{H}$. For every individual triplet, we search on $\mathcal{H}$ to encode the context information of the data distribution for the optimization objective. Details of training process with the proposed hierarchical triplet loss are described in Algorithm 1.

Hierarchical Triplet Loss

- Hierarchical Triplet Loss
  - Training architecture

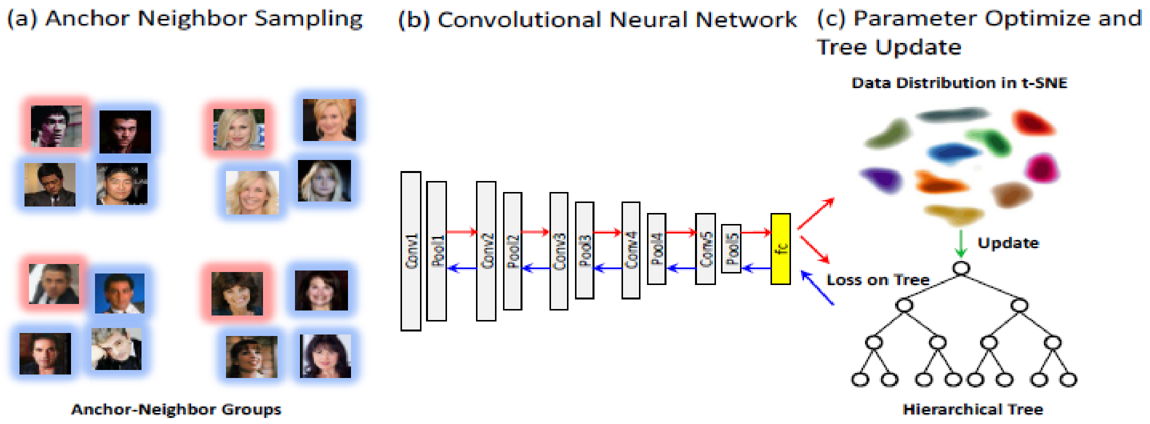
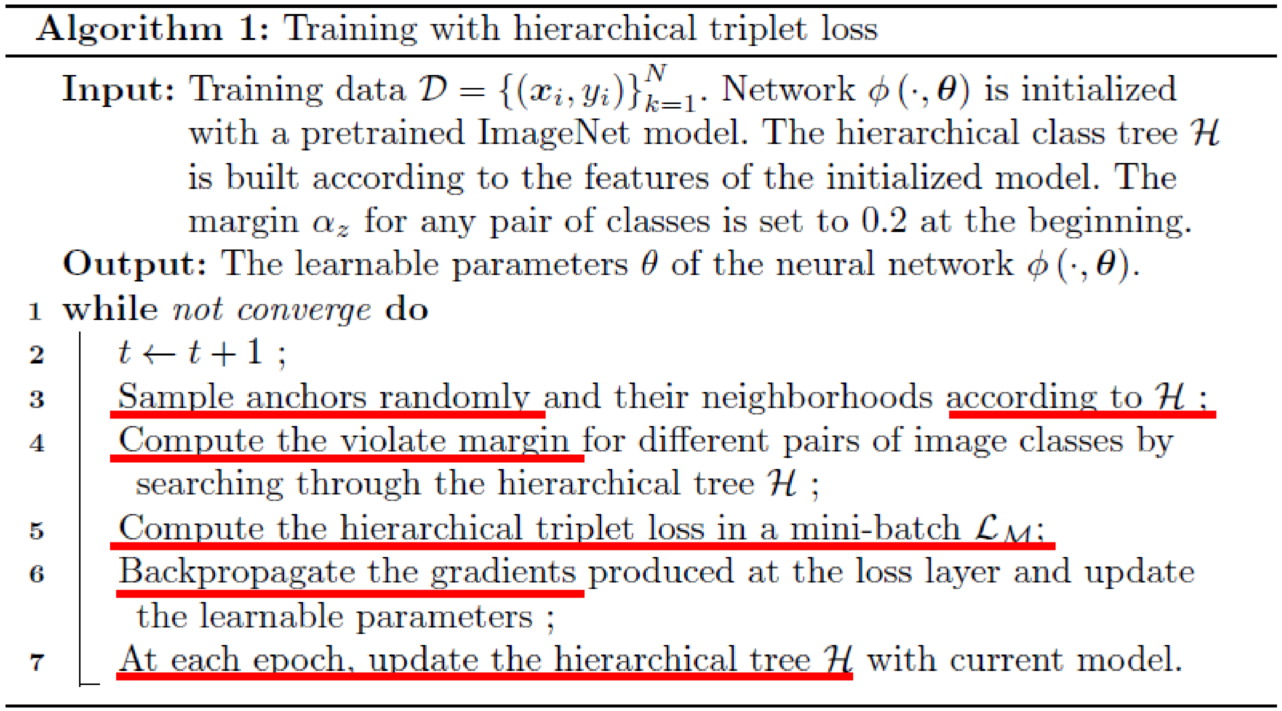

(a) Anchor Neighbor Sampling
(b) Convolutional Neural Network
(c) Parameter Optimize and Tree Update
Data Distribution in t-SNE
Conv1
Pool1
Conv2
Pool2
Conv3
Pool3
Conv4
Pool4
Conv5
Pool5
fc
Update
Loss on Tree
Anchor-Neighbor Groups
Hierarchical Tree

Fig. 3. (a)Sampling strategy of each mini-batch. The images in red stand for anchors and the images in blue stand for the nearest neighbors. (b) Train CNNs with the hierarchical triplet loss. (c) Online update of the hierarchical tree.

Hierarchical Triplet Loss

- Implementation Details.



Algorithm 1: Training with hierarchical triplet loss

```
Input: Training data $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{k=1}^{N}$. Network $\phi(\cdot, \boldsymbol{\theta})$ is initialized
       with a pretrained ImageNet model. The hierarchical class tree $\mathcal{H}$
       is built according to the features of the initialized model. The
       margin $\alpha_z$ for any pair of classes is set to 0.2 at the beginning.
Output: The learnable parameters $\theta$ of the neural network $\phi(\cdot, \boldsymbol{\theta})$.
1 while not converge do
2     $t \leftarrow t + 1$ ;
3     Sample anchors randomly and their neighborhoods according to $\mathcal{H}$ ;
4     Compute the violate margin for different pairs of image classes by
        searching through the hierarchical tree $\mathcal{H}$ ;
5     Compute the hierarchical triplet loss in a mini-batch $\mathcal{L}_{\mathcal{M}}$;
6     Backpropagate the gradients produced at the loss layer and update
        the learnable parameters ;
7     At each epoch, update the hierarchical tree $\mathcal{H}$ with current model.
```

Experimental Results and Comparisons

- In-Shop Clothes Retrieval
- Caltech-UCSD Birds 200-2011
- Cars-196 and Stanford Online Products
- LFW Face Verification
- Sampling Matter and Local Optima
- Ablation Study

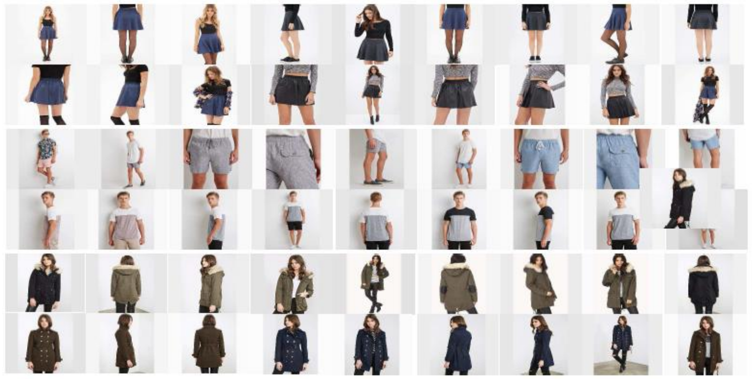
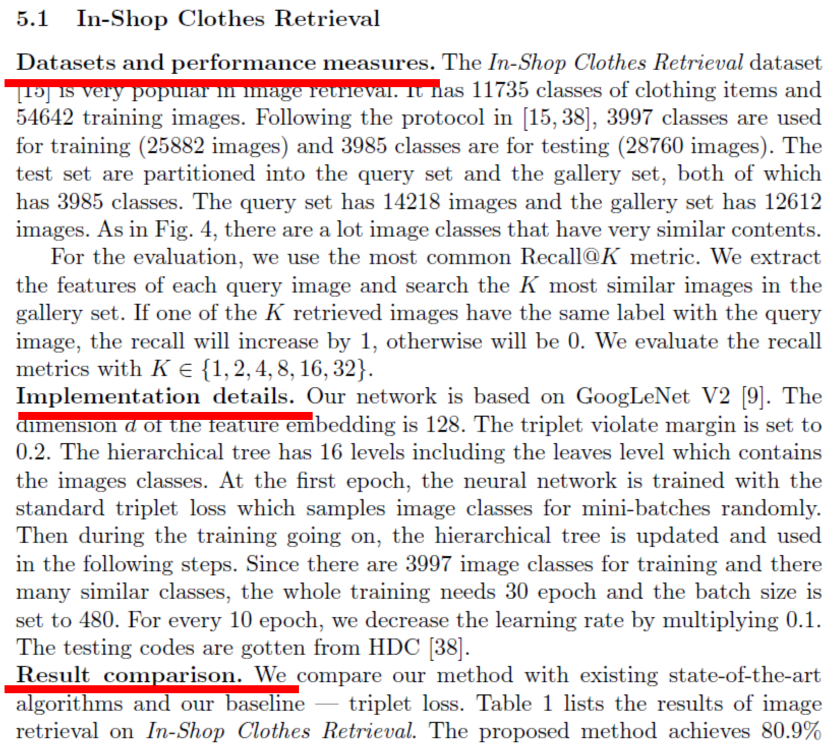


Fig. 4. Anchor-Neighbor visualization on *In-Shop Clothes Retrieval* training set [15]. Each row stands for a kind of fashion style. The row below each odd row is one of neighborhoods of the fashion style in the odd row.



5.1 In-Shop Clothes Retrieval

Datasets and performance measures. The *In-Shop Clothes Retrieval* dataset [15] is very popular in image retrieval. It has 11735 classes of clothing items and 54642 training images. Following the protocol in [15, 38], 3997 classes are used for training (25882 images) and 3985 classes are for testing (28760 images). The test set are partitioned into the query set and the gallery set, both of which has 3985 classes. The query set has 14218 images and the gallery set has 12612 images. As in Fig. 4, there are a lot image classes that have very similar contents.

For the evaluation, we use the most common Recall@K metric. We extract the features of each query image and search the K most similar images in the gallery set. If one of the K retrieved images have the same label with the query image, the recall will increase by 1, otherwise will be 0. We evaluate the recall metrics with $K \in \{1, 2, 4, 8, 16, 32\}$.

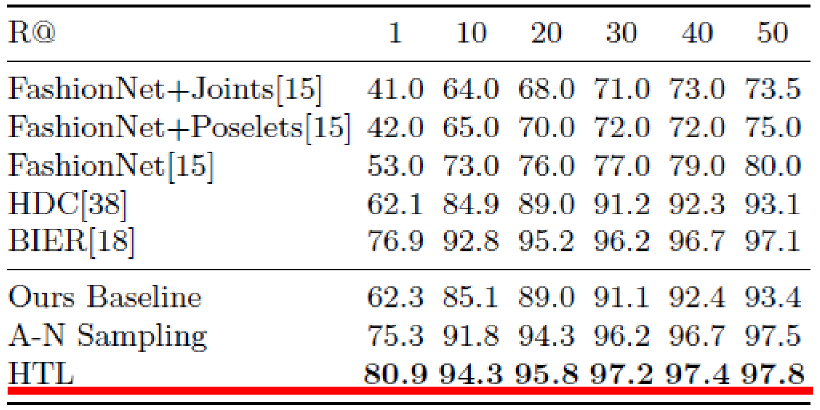
Implementation details. Our network is based on GoogLeNet V2 [9]. The dimension d of the feature embedding is 128. The triplet violate margin is set to 0.2. The hierarchical tree has 16 levels including the leaves level which contains the images classes. At the first epoch, the neural network is trained with the standard triplet loss which samples image classes for mini-batches randomly. Then during the training going on, the hierarchical tree is updated and used in the following steps. Since there are 3997 image classes for training and there many similar classes, the whole training needs 30 epoch and the batch size is set to 480. For every 10 epoch, we decrease the learning rate by multiplying 0.1. The testing codes are gotten from HDC [38].

Result comparison. We compare our method with existing state-of-the-art algorithms and our baseline — triplet loss. Table 1 lists the results of image retrieval on *In-Shop Clothes Retrieval*. The proposed method achieves 80.9%

Experimental Results and Comparisons

- In-Shop Clothes Retrieval
  - R@1, R@10 … R@50

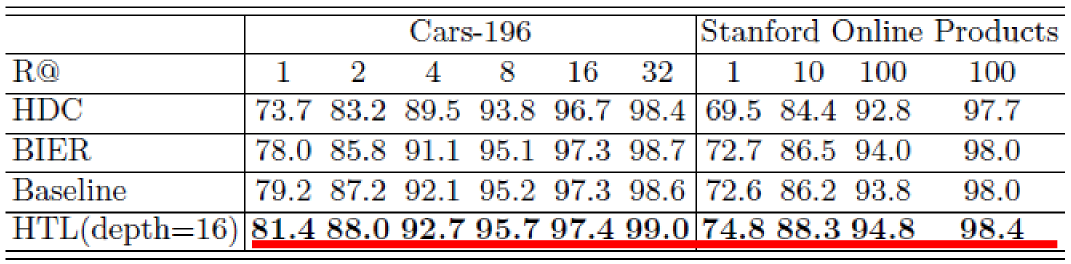
Table 1. Comparisons on the In-Shop Clothes Retrieval dataset [15].



R@	1	10	20	30	40	50
FashionNet+Joints[15]	41.0	64.0	68.0	71.0	73.0	73.5
FashionNet+Poselets[15]	42.0	65.0	70.0	72.0	72.0	75.0
FashionNet[15]	53.0	73.0	76.0	77.0	79.0	80.0
HDC[38]	62.1	84.9	89.0	91.2	92.3	93.1
BIER[18]	76.9	92.8	95.2	96.2	96.7	97.1
Ours Baseline	62.3	85.1	89.0	91.1	92.4	93.4
A-N Sampling	75.3	91.8	94.3	96.2	96.7	97.5
HTL	**80.9**	**94.3**	**95.8**	**97.2**	**97.4**	**97.8**

- Cars-196 & Stanford Online Products

Table 3. Comparison with the state-of-art on the cars-196 and Stanford products.



	Cars-196						Stanford Online Products			
R@	1	2	4	8	16	32	1	10	100	100
HDC	73.7	83.2	89.5	93.8	96.7	98.4	69.5	84.4	92.8	97.7
BIER	78.0	85.8	91.1	95.1	97.3	98.7	72.7	86.5	94.0	98.0
Baseline	79.2	87.2	92.1	95.2	97.3	98.6	72.6	86.2	93.8	98.0
HTL(depth=16)	**81.4**	**88.0**	**92.7**	**95.7**	**97.4**	**99.0**	**74.8**	**88.3**	**94.8**	**98.4**

Experimental Results and Comparisons

- Sampling Matter and Local Optima
  - They investigate the influence of batch size on the test set of In-Shop Clothes Retrieval.
  - Fig. 5 (a) shows that when the batch size grows from 60 to 480, the accuracy increases in the same iterations.

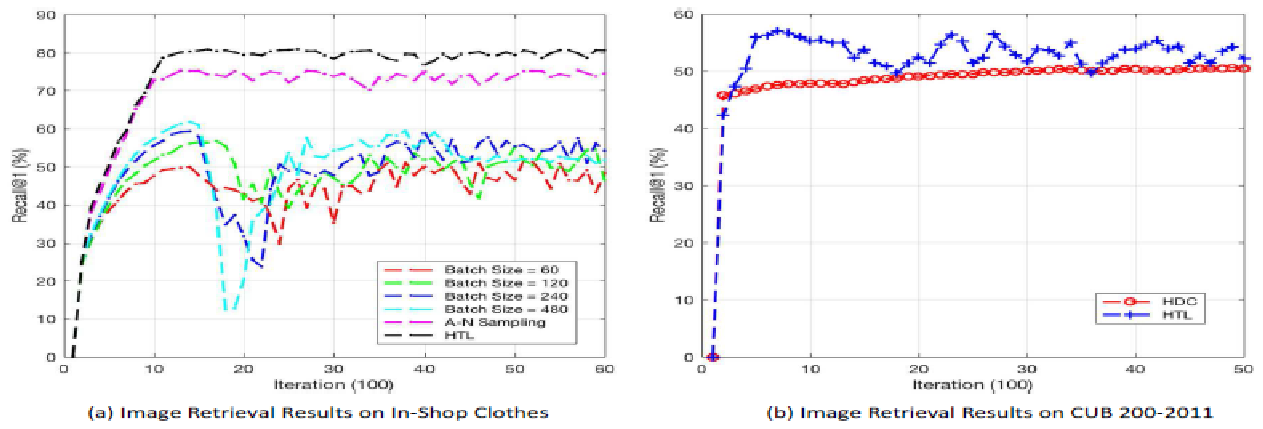


Fig. 5. (a) Image retrieval results on In-Shop Clothes [15] with various batch sizes. (b) Image retrieval results on CUB-200-2011 [31].

Conclusion

- hierarchical triplet loss(HTL) is able to select informative training samples (triplets) via an adaptively-updated hierarchical tree that encodes global context.
- HTL effectively handles the main limitation of random sampling, which is a critical issue for deep metric learning.
- First, They construct a hierarchical tree at the class level which encodes global context information over the whole dataset. → Visual similar classes are merged recursively to form the hierarchy.
- Second, the problem of triplet collection is formulated by proposing a new violate margin, which is computed dynamically based on the designed hierarchical tree.
- This allows it to learn from more meaningful hard samples with the guide of global context.
- The proposed HTL is evaluated on the tasks of image retrieval and face recognition, where it achieves new state-of-the-art performance on a number of standard benchmarks.