Hierarchical Proxy-based Loss for Deep Metric Learning

Zhibo Yang[1*], Muhammet Bastan[2], Xinliang Zhu[2], Doug Gray[2], Dimitris Samaras[1]
[1]Stony Brook University, [2]Visual Search & AR, Amazon

Abstract

Proxy-based metric learning losses are superior to pair-based losses due to their fast convergence and low training complexity. However, existing proxy-based losses focus on learning class-discriminative features while overlooking the commonalities shared across classes which are potentially useful in describing and matching samples. Moreover, they ignore the implicit hierarchy of categories in real-world datasets, where similar subordinate classes can be grouped together. In this paper, we present a framework that leverages this implicit hierarchy by imposing a hierarchical structure on the proxies and can be used with any existing proxy-based loss. This allows our model to capture both class-discriminative features and class-shared characteristics without breaking the implicit data hierarchy. We evaluate our method on five established image retrieval datasets such as In-Shop and SOP. Results demonstrate that our hierarchical proxy-based loss framework improves the performance of existing proxy-based losses, especially on large datasets which exhibit strong hierarchical structure.

1. Introduction

Learning visual similarity has many important applications in computer vision, ranging from image retrieval [10] to video surveillance (e.g., person re-identification [5]). It is often treated as a metric learning problem where the task is to represent images with compact embedding vectors such that semantically similar images are grouped together while dissimilar images are far apart in the embedding space. Inspired by the success of deep neural networks in visual recognition, convolutional neural networks have also been employed in metric learning, which is specifically called deep metric learning (DML) [2, 4, 5, 8, 19, 24, 25, 41].

In recent years, a number of DML loss functions [17, 22, 32] have been developed to guide network training for visual similarity learning. The two dominant groups of these DML loss functions are pair-based losses and proxy-based losses. Pair-based losses (e.g., contrastive loss [11] and triplet loss [32]) directly compute the loss based on pairs of samples with the goal of encouraging samples in the same class to be close to each other and samples from different classes to be far apart. This is different from classification losses which are computed individually for each training sample. Pair-based losses compute loss for each tuple of samples formed in a training batch. Hence, a major drawback of pair-based losses is that for a fixed number N of training samples there is a prohibitively large number of tuples (i.e., $O(N^2)$ or $O(N^3)$) including many non-informative tuples, which leads to slow convergence and degraded performance. In contrast, proxy-based losses (e.g., proxy-NCA [22] and proxy anchor loss [17]) try to learn a set of data points, called *proxies*, to approximate the data space of the training set. At each iteration, triplets are formed between samples from a local training batch and the global proxies to train the embedding networks as well as to update the proxies. Since the number of proxies is often

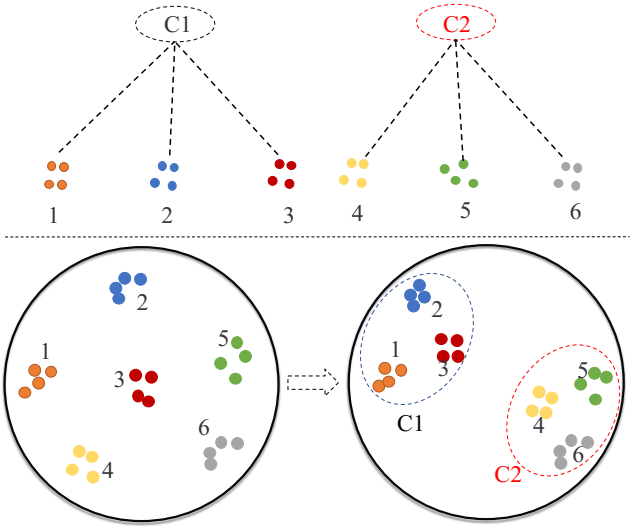

Figure 1. **Hierarchical proxies**. Traditional proxy-based losses seek to learn an embedding space where each class is well separated (see bottom-left where different colors denote different classes). However, real-world data often have implicit hierarchies. For example, classes 1-3 might belong to one super category while classes 4-6 belong to another (top panel). Our proposed HPL learns to separate the classes and captures this hierarchy to pull samples in the same super category closer (bottom-right).

*Work partially done while interning at Amazon.

orders of magnitude smaller than the number of samples in the training set, proxy-based losses significantly reduce the high training complexity of pair-based losses.

In the image retrieval context, where the task is to find visually similar images in a large gallery of seen and unseen classes given a query image, real-world datasets, like SOP [25], ImageNet [30], often contain an implicit hierarchy of categories, e.g., huskies can be categorized as spitzs, as simply dogs, or more broadly as animals. Common features shared across the finer-grained classes often characterize their superordinate category. However, existing proxy-based losses ignore this hierarchy, focusing on extracting class-discriminative features and overlooking features that are shared across classes that could be useful for describing and retrieving images. Taking dogs as an example, collies and German shepherds are medium-sized dogs with thin snouts and upright ears, while collies are distinctive from German shepherds in their long hair. Given a training set of collie and German shepherd images, a typical proxy-based model might only learn features related to long hair while overlooking features shared among classes such as medium size which is helpful in discerning collies from shelties.

In this paper, we propose a simple method that imposes a hierarchical structure on the proxies (see Fig. 1 for illustration) and can be combined with existing proxy-based losses. We name our method *Hierarchical Proxy-based Loss (HPL)*. Building on top of existing proxy-based losses, our HPL creates a pyramid of learnable proxies where the lowest/finest level of proxies are created similarly to the existing proxy-based losses (i.e, one proxy per class). The higher-level proxies are learned in an unsupervised manner. Each coarser level of proxies are the cluster centroids of its lower level of proxies. Hence, each sample is associated with one proxy at every level in the proxy pyramid and the losses are computed for each level independently. In experiments, we demonstrate that our HPL improves traditional proxy-based loss performance on several public image retrieval benchmarks including Stanford Online Products (SOP) and In-Shop Clothes Retrieval (In-Shop), CUB200, Cars-196 and iNaturalist. Verifying our initial motivation, larger performance improvements are observed for datasets with a larger number of classes and a more complex hierarchical structure—Recall@1 is increased by +2.50% on SOP and +2.87% on In-Shop. Performance still matches the state-of-the-art in the case of the CUB200 dataset which contains 200 fine-grained bird categories with subtle differences, lacking a strong hierarchy.

In summary, our contributions are: 1) We propose a novel hierarchical proxy-based method which is applicable to all existing proxy-based losses; 2) We demonstrate that the proposed method improves the performance of traditional proxy-based losses on several image retrieval benchmarks, especially for datasets with strong and complex hierarchical structure; 3) We also show that the hierarchy learned by our method outperforms a human-curated hierarchy for image retrieval.

2. Related Work

Proxy-based losses. Proxy-NCA [22], the first proxy-based DML loss, addressed the slow convergence of pair-based losses (e.g. Triplet loss [32]). The main idea is to create one proxy for each class (by default) and use the Neighborhood Component Analysis (NCA) loss [28] to pull samples close to its assigned proxy while pushing samples away from non-assigned proxies. Proxy-NCA++ [34] further improves Proxy-NCA with several training improvements. Proxy Anchor [17] uses proxies as anchors to leverage the rich image-to-image relations that are missing in Proxy-NCA. Other methods like SoftTriple loss [26] and Manifold Proxy loss [1] respectively extend the Softmax and N-pair losses to proxy-based versions. To an extent, all these proxy-based losses treat deep metric learning as a classification task by using the proxies to separate samples from different classes, and thus they focus on extracting class-discriminative features. In contrast, our proposed HPL builds on existing proxy-based losses and explicitly models the class-shared features by imposing a hierarchical structure in the proxies. This helps to regularize the embeddings and prevent the model from overfitting to the training classes.

Modeling the data distribution. Similar to our approach, hierarchical triplet loss (HTL) [9] and HiBsteR [36] also try to leverage the underlying data hierarchy for DML. However, HTL has been developed for the Triplet loss and uses the data hierarchy to mine good training samples, while our method is designed for proxy-based losses and aims to learn class-shared information by modeling the data hierarchy. HiBsteR requires ground-truth hierarchical labels which limits its applicability, while our method does not. Divide and Conquer (DC) [31] tries to adapt the embeddings to a nonuniform data distribution, while we aim to learn embeddings that capture the underlying data hierarchy. Moreover, DC needs to cluster the entire dataset repeatedly which can be prohibitively expensive for large datasets, while our method operates on proxies which are often orders of magnitude smaller than the whole dataset. In addition, our HPL also resembles PIEs [13] in learning from different groups of data, but PIEs aims to learn a pose-invariant embedding from different views of objects in each class and requires additional pose labels while our HPL learns from groups of classes and does not require additional annotation.

Modeling class-shared information. MIC [27] and DiVA [21] also try to learn features shared across classes. However, both MIC and DiVA formulate a multi-task learning problem and seek to learn separate embeddings for class-

specific information and class-shared information. Instead, our HPL imposes additional regularization on top of the original proxy-based losses by explicitly modeling class-shared information. Moreover, MIC clusters over the entire dataset which is not scalable to large datasets, while we cluster on the proxies whose number is much smaller than the size of the entire dataset. DiVA is based on triplet loss while our HPL is developed for proxy-based losses and is compatible with multiple proxy-based losses. Our HPL also shares a similar motivation with another line of research [16, 23, 42] which aims to design better image classifiers by leveraging the hierarchical structure in data. However, they focus on the network architectures design to better separate the classes, while our work aims to design a better metric learning loss function which takes advantages of class-shared information.

3. Method

In this section, we briefly review two popular proxy-based losses: Proxy-NCA and Proxy Anchor. Then we introduce our hierarchical proxy-based loss.

3.1. Preliminaries

Given a training set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ where N is the number of data samples, x_i and y_i are the i-th training image and its class label, respectively. The goal of deep metric learning is to learn a similarity function $s(x_i, x_j)$ such that

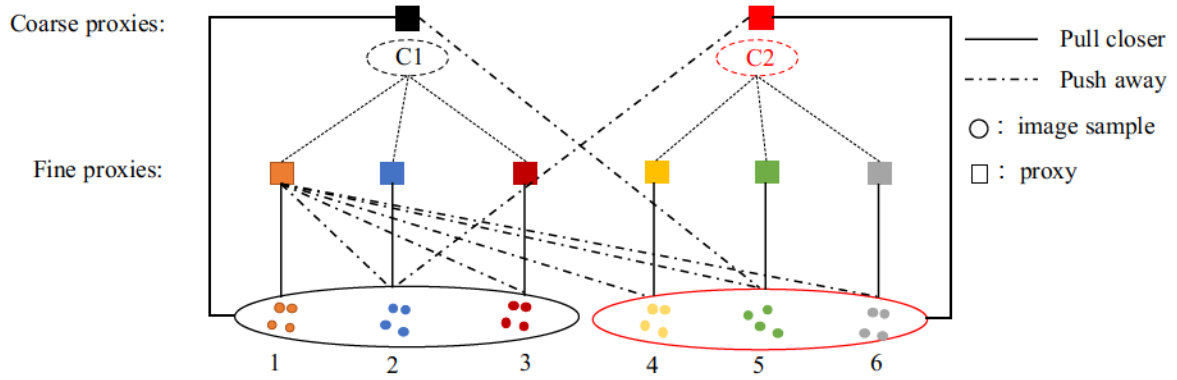

Figure 2. **The framework of our hierarchical proxy-based loss**. HPL builds on top of existing proxy-based losses and computes the loss for each level of proxies separately. HPL learns an embedding space with small within-class distance but also small within-cluster (i.e., coarse proxies) distance. For illustration clarity, the full interactions between fine proxies and samples are presented for only class 1, class 2-6 are similar. Each color represent a category or super category.

Note that Eq. (5) is a generalization of Eq. (2) and is equivalent to Eq. (2) when $L = 1$. Thus, we extend existing proxy-based losses and formulate the HPL loss as:

$\arg\min_j ||P_l^i - P_{l+1}^j||_2^2$. Then, we update the higher-level proxies by taking the average of the lower-level proxies that are assigned to them, i.e., $P_{l+1}^j = \sum_{Q_l^i=j} P_l^i / |\mathbf{1}_{Q_l=j}|$, where $\mathbf{1}_{Q_l=j}$ is a vector with all ones at the location where $Q_l^i = j$ and zeros elsewhere. Note that our method can be easily adapted to work with other clustering algorithms such as Gaussian Mixture models [43].

4. Experiments

To evaluate the proposed hierarchical proxy-based loss (HPL), we follow both a recently proposed standardized evaluation protocol proposed in [24] and the traditional evaluation protocol as in [17, 22]. We compare our HPL against existing proxy-based losses including Proxy-NCA [22] and Proxy Anchor [17] on several public benchmark datasets on image retrieval. We denote our HPL implemented based on Proxy-NCA loss and Proxy Anchor loss by HPL-NCA and HPL-PA, respectively. In addition, we also study the impact of different hyper-parameters to the performance of HPL as well as the effectiveness of the online clustering module of HPL. Recall@K, Mean Average Precision at R (MAP@R) [24] and R-precision (RP) are used to measure the image retrieval performance

4.1. Datasets

Five popular benchmark datasets are used to evaluate our method: In-shop Clothes Retrieval (In-Shop) [20], Stanford Online Products (SOP) [25], CUB-200-2011 (CUB) [40], Cars-196 [18] and iNaturalist [14]. **In-Shop** [20] contains 72,712 clothes images of 7,986 classes, among which, the first 3,997 classes are used for training and the remaining 3985 classes for testing. Note that the testing data is split into a query set and a gallery set which contain 14,218 images and 12,612 images, respectively. **SOP** [25] consists of 120,053 online product images of 22,634 classes and 24 super classes. We use the first 59,551 images (11,318 classes, 12 super classes) for training and the remaining 60,502 (11,316 classes, 12 super classes) for testing. **Cars-196** [18] is composed of 196 car models (i.e., classes) with 16,185 images. We train the models on the first 98 classes (8,054 images) and test on the remaining 100 classes (8,131 images). **CUB-200-2011** [40] contains 11,788 images of 200 bird species (i.e., classes). We train the models on the first 100 classes (5,864 images) and test on the remaining 100 classes (5,924 images). **iNaturalist** [14] is a fine-grained dataset of animal and plant species with human-curated hierarchy of categories. We use iNaturalist 2019[2] which contains 1,010 species, spanning 72 genera, combining a total of 268,243 images in the training and validation set. Each genus contains at least 10 species, making the dataset well-balanced in its category hierarchy. We follow [3] and use the first 656 species (48 genera) for training and the remaining 354 species (24 genera) for testing. We will make the train/test splits publicly available.

[2]We choose iNaturalist 2019 over iNaturalist 2018 because the former is more balanced in its category hierarchy.

4.2. Implementation Details

Embedding network. We use the Inception network with batch normalization (BN-Inception) [15] and ResNet-50 [12] as the backbone networks. Both backbone networks are pretrained on ImageNet [6] for the classification task. We append a max pooling layer in parallel with an average pooling layer to the penultimate layer of the backbone network as in [17] and replace the final fully-connected (fc) layer of the backbone network with a new fc layer (i.e., embedding layer) which projects the network output to a desired embedding space.

Structure of the hierarchical proxies. The structure of the hierarchical proxies varies for different datasets as different datasets have different hierarchical characteristics. However, to keep the analysis simple, we evaluate our model in a simple conceptual structure, a two-level hierarchy. Namely, the number of fine (i.e, lower-level) proxies is set as the number of classes in the dataset. While the coarse (higher-level) proxies is chosen differently for different datasets as the number of classes in each dataset varies significantly. Please find the detailed study of the impact of the hierarchical structure at Sec. 4.5.

Training setting. Different from HTL [9] which initializes the hierarchical tree by clustering the whole datasets, which is expensive and infeasible for large datasets, we train the finest level of proxies P_0 and the embedding network with standard proxy losses (i.e., Proxy-NCA or Proxy Anchor) for the first 3 epochs to prevent the model from capturing wrong data hierarchy during the early stage of training. The higher level of proxies are updated at every epoch. The loss weights in Eq. (6) are set as $\omega_0 = 1.0$ and $\omega_1 = 0.1$. Under the standardized evaluation protocol [24] all hyper-parameters of the models are determined using Bayesian Optimization [33] and cross validation and the training terminates once the validation error plateaus. The embedding size is set to 128 and BN-Inception is used as the backbone networks. During testing, two modes are used for evaluation: *Concatenated* where 128-dim embeddings of the 4 models trained with 4-fold cross validation are concatenated (512-dim) and *Separated* where the 4 models are evaluated separately (128-dim) and the average performance is reported. Under the traditional evaluation protocol, we train the baseline models using ResNet-50 as backbone with both Proxy-NCA loss and Proxy Anchor loss by following the standard hyper-parameters settings in [22] and [17], respectively. Each model is trained for 30 epochs with learning rate 10^{-4} and batch size 128. The embedding dimension is

	Concatenated (512-dim)			Separated (128-dim)		
	MAP@R	P@1	RP	MAP@R	P@1	RP
Contrastive [11]	44.51 ± 0.28	73.27 ± 0.23	47.45 ± 0.28	40.29 ± 0.27	69.28 ± 0.22	43.39 ± 0.28
CosFace [37]	46.92 ± 0.19	75.79 ± 0.14	49.77 ± 0.19	40.69 ± 0.21	70.71 ± 0.14	43.56 ± 0.21
ArcFace [7]	$\mathbf{47.41 \pm 0.40}$	$\mathbf{76.20 \pm 0.27}$	$\mathbf{50.27 \pm 0.38}$	41.11 ± 1.22	$\mathbf{70.88 \pm 1.51}$	44.00 ± 1.26
MS [38]	46.42 ± 1.67	75.01 ± 1.21	49.45 ± 1.67	$\mathbf{41.24 \pm 1.89}$	70.65 ± 1.70	$\mathbf{44.40 \pm 1.85}$
SoftTriple [26]	47.35 ± 0.19	76.12 ± 0.17	50.21 ± 0.18	40.92 ± 0.20	70.88 ± 0.20	43.83 ± 0.20
Proxy-NCA++ [34]	46.56	75.10	49.50	41.51	70.43	43.82
Proxy-NCA [22]	47.22 ± 0.21	75.89 ± 0.17	50.10 ± 0.22	41.74 ± 0.21	71.30 ± 0.20	44.71 ± 0.21
HPL-NCA	**47.97**	**76.60**	**50.87**	**42.06**	**71.77**	**45.06**
Proxy Anchor [17]	47.88	76.12	50.82	43.97	72.79	47.00
HPL-PA	**49.07**	**76.97**	**51.97**	**45.11**	**73.84**	**48.10**

Table 1. **Results on the SOP dataset**. All hyper-parameters are selected by Bayesian Optimization as in [24]. Average performance across 10 runs and the 95% confidence interval are reported whenever applicable. The best numbers in each block are marked as bold and the best numbers in the table are highlighted in blue. Cont. + XBM is not included because it failed to converge under this training setting.

	Concatenated (512-dim)			Separated (128-dim)		
	MAP@R	P@1	RP	MAP@R	P@1	RP
Contrastive [11]	25.49 ± 0.41	81.57 ± 0.36	35.72 ± 0.35	17.61 ± 0.24	69.44 ± 0.24	28.15 ± 0.21
CosFace [37]	26.86 ± 0.22	85.27 ± 0.23	36.72 ± 0.20	18.22 ± 0.11	$\mathbf{74.13 \pm 0.21}$	28.49 ± 0.14
ArcFace [7]	27.22 ± 0.30	$\mathbf{85.44 \pm 0.28}$	37.02 ± 0.29	17.11 ± 0.18	72.10 ± 0.37	27.29 ± 0.17
MS [38]	$\mathbf{27.84 \pm 0.77}$	85.29 ± 0.31	$\mathbf{37.96 \pm 0.63}$	$\mathbf{18.77 \pm 0.69}$	73.73 ± 0.96	29.38 ± 0.60
SoftTriple [26]	26.06 ± 0.19	83.66 ± 0.22	36.31 ± 0.16	18.72 ± 0.11	72.98 ± 0.16	$\mathbf{29.39 \pm 0.10}$
Cont. + XBM [39]	26.04 ± 0.24	83.67 ± 0.35	36.10 ± 0.19	18.07 ± 0.11	72.58 ± 0.21	28.55 ± 0.10
Proxy-NCA++ [34]	26.02 ± 0.26	82.09 ± 0.41	36.31 ± 0.24	18.63 ± 0.09	70.60 ± 0.18	29.35 ± 0.08
Proxy-NCA [22]	25.56 ± 0.15	83.20 ± 0.22	35.80 ± 0.12	18.32 ± 0.12	$\mathbf{73.34 \pm 0.13}$	28.87 ± 0.11
HPL-NCA	$\mathbf{27.47 \pm 0.20}$	$\mathbf{84.54 \pm 0.25}$	$\mathbf{37.56 \pm 0.20}$	$\mathbf{18.87 \pm 0.13}$	72.27 ± 0.18	$\mathbf{29.45 \pm 0.15}$
Proxy Anchor [17]	27.77 ± 0.20	86.38 ± 0.15	37.53 ± 0.17	19.82 ± 0.10	$\mathbf{76.85 \pm 0.13}$	30.12 ± 0.10
HPL-PA	$\mathbf{28.67 \pm 0.22}$	$\mathbf{86.84 \pm 0.31}$	$\mathbf{38.36 \pm 0.18}$	$\mathbf{19.83 \pm 0.10}$	76.12 ± 0.34	$\mathbf{30.13 \pm 0.09}$

Table 2. **Results on the Cars-196 dataset**. All hyper-parameters are selected by Bayesian Optimization as in [24]. Average performance across 10 runs and the 95% confidence interval are reported.

	Concatenated (512-dim)			Separated (128-dim)		
	MAP@R	P@1	RP	MAP@R	P@1	RP
Contrastive [11]	26.19 ± 0.28	67.21 ± 0.49	36.92 ± 0.28	20.73 ± 0.19	58.63 ± 0.46	31.48 ± 0.19
CosFace [37]	26.53 ± 0.23	67.19 ± 0.37	37.36 ± 0.23	21.25 ± 0.18	59.83 ± 0.30	32.07 ± 0.19
ArcFace [7]	26.45 ± 0.20	67.50 ± 0.25	37.31 ± 0.21	21.49 ± 0.16	60.17 ± 0.32	32.37 ± 0.17
MS [38]	25.16 ± 0.10	65.93 ± 0.16	35.91 ± 0.11	20.58 ± 0.09	58.51 ± 0.18	31.36 ± 0.10
SoftTriple [26]	25.64 ± 0.21	66.20 ± 0.37	36.46 ± 0.20	21.26 ± 0.18	59.55 ± 0.35	32.10 ± 0.19
Cont. + XBM [39]	$\mathbf{26.85 \pm 0.63}$	$\mathbf{68.43 \pm 1.18}$	$\mathbf{37.66 \pm 0.56}$	$\mathbf{21.78 \pm 0.35}$	$\mathbf{60.95 \pm 0.76}$	$\mathbf{32.69 \pm 0.33}$
Proxy-NCA++ [34]	23.53 ± 0.12	64.69 ± 0.40	34.37 ± 0.13	18.76 ± 0.15	57.13 ± 0.36	29.52 ± 0.16
Proxy-NCA [22]	23.85 ± 0.24	65.01 ± 0.27	34.79 ± 0.26	19.15 ± 0.15	$\mathbf{57.49 \pm 0.35}$	29.99 ± 0.15
HPL-NCA	$\mathbf{24.95 \pm 0.21}$	$\mathbf{65.22 \pm 0.23}$	$\mathbf{35.70 \pm 0.21}$	$\mathbf{20.04 \pm 0.21}$	57.45 ± 0.13	$\mathbf{30.79 \pm 0.21}$
Proxy Anchor [17]	26.47 ± 0.21	67.64 ± 0.42	37.29 ± 0.19	21.57 ± 0.15	60.59 ± 0.24	32.45 ± 0.15
HPL-PA	$\mathbf{26.72 \pm 0.18}$	$\mathbf{68.25 \pm 0.29}$	$\mathbf{37.57 \pm 0.18}$	$\mathbf{21.90 \pm 0.19}$	$\mathbf{61.31 \pm 0.25}$	$\mathbf{32.81 \pm 0.19}$

Table 3. **Results on the CUB dataset**. All hyper-parameters are selected by Bayesian Optimization as in [24]. Average performance across 10 runs and the 95% confidence interval are reported.

	In-Shop		SOP	
	R@1	R@10	R@1	R@10
Proxy-NCA	87.21	96.57	77.63	89.29
HPL-NCA	**88.70**	**96.83**	**80.13**	**91.07**
Proxy Anchor	89.85	97.14	79.38	90.44
HPL-PA	**92.46**	**97.97**	**80.04**	**91.05**

Table 4. **Recall@K (%) on the In-Shop and SOP datasets**. ResNet-50 is used as the backbone and we set $|P_1| = 500$ for both In-Shop and SOP.

set to 512. More details can be found in the supplement.

4.3. Comparing Deep Metric Learning Models

Main Results. To eliminate the bias incurred by the choice of hyper-parameters, we follow a stringent evaluation protocol proposed in [24] and compare our method based on Proxy Anchor loss (HPL-PA) with existing metric learning losses including Contrastive [11], CosFace [37], ArcFace [7], MultiSimilarity (MS) [38], SoftTriple [26], Contrastive with Cross-Batch Memory (Cont. + XBM) [39] and Proxy-NCA++ [34]. Table 1-3 shows the MAP@R, P@1 (i.e., Recall@1) and RP on the SOP, Cars-196 and CUB datasets, respectively. The results demonstrate the effectiveness of our proposed HPL. One can see that our HPL consistently improves both Proxy-NCA loss and Proxy Anchor loss in most metrics across all three datasets. Observe that HPL improves the MAP@R of Proxy-NCA from 25.56% to 27.47% and from 47.22% to 47.92% on the Cars-196 and SOP datasets, respectively, making Proxy-NCA comparable with or even better than a recent proxy-based loss—Proxy Anchor. This suggests that the improvement of HPL over Proxy-NCA is significant. Comparing with Proxy Anchor, our HPL-PA boosts the MAP@R by 1.19%, 0.9%, and 0.23% on SOP, Cars-196 and CUB, respectively. This could imply that HPL tends to work better on large datasets with stronger data hierarchy (e.g., SOP) than on small ones (e.g., CUB and Cars-196) which lack hierarchy. Moreover, HPL-PA achieves state-of-the-art performance on all three datasets comparing with all other approaches in most of the scenarios. In the CUB dataset, our HPL-PA is slightly worse than XBM with Contrastive loss, but HPL-PA is much more stable than XBM as one can see from the confidence interval.

We further compare our method with ResNet-50 as the backbone on two large datasets: In-Shop and SOP, using the traditional evaluation protocol. The number of coarse proxies in our HPL for the In-Shop dataset and SOP dataset are set to 500. The results in Table 4 show that our HPL improves the traditional Proxy-NCA and Proxy Anchor on both benchmarks consistently. Especially, HPL-PA surpasses Proxy Anchor by 2.87% on the In-Shop dataset and

Figure 3. **Qualitative results on SOP**. Odd rows are the results of our HPL-NCA loss; even rows are the results from the Proxy-NCA loss. Green box indicates correct match, while red box indicates wrong match.

HPL-NCA outperforms Proxy-NCA by 2.50% on the SOP dataset in Recall@1. This implies that the inclusion of class-shared information boosts the image retrieval performance and our HPL helps to capture this information as well as the class-discriminative features. More results can be found in the supplement.

Qualitative analysis. To further evaluate our method, we qualitatively compare our HPL-NCA with the Proxy-NCA loss by presenting the image retrieval results. In Fig. 3, we present the rank-1, 2, 31, and 32 retrieval results on SOP to illustrate the improvement on the embeddings quality. One can see our HPL-NCA is better than Proxy-NCA in both overall quality of the image retrieval results and accuracy. Specifically, Fig. 3 reveals that despite both methods finding the correct matches, the retrieval results from our method are more similar to the queries. For example, on the left panel of Fig. 3 Proxy-NCA returns a yellow kettle as the 31th match, while our HPL-NCA returns a brown wood cabinet which is much more similar to the query image—a yellow wood hutch. Please find more qualitative results in the supplement.

Furthermore, in Fig. 4 we visualize the embeddings of the test set of Cars-196 (98 classes) learned by our HPL-NCA (10 coarse proxies) and Proxy-NCA with t-SNE [35]. As highlighted in red boxes, our method groups similar car categories (e.g., pickup trucks) into a larger cluster, while still maintaining good separability between classes. The learned common truck features help discern trucks from unseen car categories like SUVs.

4.4. Learning with a Human-curated Hierarchy

Our method learns the data hierarchy in an unsupervised manner (see Sec. 3.4 for details). To demonstrate its effec-

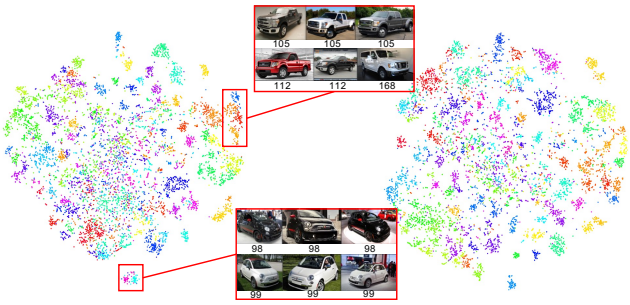

Figure 4. **Visualization of the embedding space.** We visualize

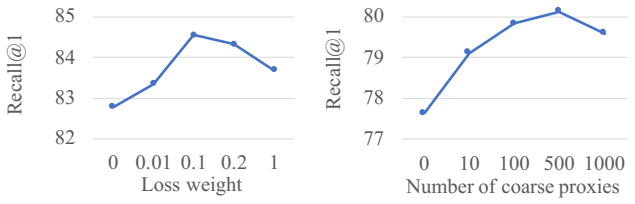

Figure 5. **Impact of loss weight (left) and number of coarse proxies (right)**. Left: Recall@1 on the Cars-196 dataset of models that are trained with 20 coarse proxies but different loss weights ω_1. Right: Recall@1 on the SOP dataset of models that are trained with different number of coarse proxies ($\omega_1 = 0.1$).

tiveness, we replace the online clustering module in Algorithm 1 with a ground-truth class hierarchy. To this end, we use the SOP and the iNaturalist datasets where human-curated hierarchical labels are available. In particular, instead of performing online clustering with the fine proxies, we use a fixed assignment of the fine proxies to coarse proxies given by the ground-truth class hierarchy. The results in Table 5 show that both HPL-NCA and HPL-NCA-GT (i.e., HPL-NCA with ground-truth hierarchy) outperform Proxy-NCA, and surprisingly, HPL-NCA surpasses HPL-NCA-GT even when given the same number of coarse proxies. This might be due to the fact that the human-curated category hierarchy may not fully reflect the visual similarity among classes, whereas, our method automatically learns the hierarchy based on visual similarity between classes, making it more favorable for metric learning. Please see the supplement for full results and further discussion.

4.5. Impact of Hyperparameters

Loss weight. In Eq. (6), we combine losses contributed by different levels of proxies using a weighted summation with the weights ω as hyper-parameters. We fix $\omega_0 = 1.0$ and train the embedding networks on Cars-196 by varying ω_1. As shown in Fig. 5, the model performs best when $\omega_1 = 0.1$. Note that when $\omega_1 = 0$ our HPL-NCA loss is equivalent to the standard Proxy-NCA loss. Hence, for all $\omega_1 > 0$, our HPL-NCA loss consistently beats the standard Proxy-NCA loss (i.e., $\omega_1 = 0$). This further demonstrates the superiority of our HPL over standard proxy-based losses.

	SOP		iNaturalist	
	$\|P_1\|$	R@1	$\|P_1\|$	R@1
Proxy-NCA	-	77.63	-	51.32
HPL-NCA-GT	12	78.69	48	51.63
HPL-NCA	12	79.33	48	51.95
HPL-NCA	500	**80.13**	500	**52.26**

Table 5. **Learned hierarchy vs human-curated hierarchy**. HPL-NCA-GT denotes PL-NCA with ground-truth class hierarchy.

Hierarchical structure. To better study the impact of the hierarchical structure, we use the SOP dataset which contains 11,316 training classes—much more than the other four datasets. Specifically, we train models on the SOP dataset using HPL-NCA loss with a variable number of coarse proxies $|P_1| = 0$, 10, 100, 500 and 1000. Note that when $|P_1| = 0$ HPL-NCA is equivalent to Proxy-NCA. The results in Fig. 5 show that our method outperforms the baseline under different numbers of coarse proxies and the performance of our method is shown to be robust with respect to the number of coarse proxies. In general, having more coarse proxies is beneficial as more accurate class-shared information can be learned from the coarse proxies. However, an excessively large number of coarse proxies would possibly reduce the strength of the signal shared across classes. More hyperparameter analysis can be found in the supplement.

5. Conclusion

In this paper, we have demonstrated the effectiveness of a hierarchical proxy-based loss (HPL), which enforces a hierarchical structure on the learnable proxies. In this way, we are able to not only learn class-discriminative information, but also capture the features that are shared across classes, which improves the generalizability of the learned embeddings. As a result, our proposed HPL improves standard proxy-based losses, especially on large datasets where a clear hierarchical structure exists in the data space. In future work, we will explore more configurations of our method, e.g., instead of using a simple two-level hierarchy and k-means where the number of clusters has to be predefined, we can use hierarchical clustering algorithms to automatically learn a multi-level hierarchy to better fits the data.

Acknowledgements. This project is supported by US National Science Foundation Award IIS-1763981, the Partner University Fund, the SUNY2020 Infrastructure Transportation Security Center, and a gift from Adobe.

References

[1] Nicolas Aziere and Sinisa Todorovic. Ensemble deep manifold similarity learning using hard proxies. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7299–7307, 2019.

[2] Malik Boudiaf, Jérôme Rony, Imtiaz Masud Ziko, Eric Granger, Marco Pedersoli, Pablo Piantanida, and Ismail Ben Ayed. A unifying mutual information view of metric learning: cross-entropy vs. pairwise losses. In *European Conference on Computer Vision*, pages 548–564. Springer, 2020.

[3] Andrew Brown, Weidi Xie, Vicky Kalogeiton, and Andrew Zisserman. Smooth-ap: Smoothing the path towards large-scale image retrieval. In *European Conference on Computer Vision*, pages 677–694. Springer, 2020.

[4] Weihua Chen, Xiaotang Chen, Jianguo Zhang, and Kaiqi Huang. Beyond triplet loss: a deep quadruplet network for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 403–412, 2017.

[5] Zuozhuo Dai, Mingqiang Chen, Xiaodong Gu, Siyu Zhu, and Ping Tan. Batch dropblock network for person re-identification and beyond. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3691–3701, 2019.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. Ieee, 2009.

[7] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019.

[8] Ismail Elezi, Sebastiano Vascon, Alessandro Torcinovich, Marcello Pelillo, and Laura Leal-Taixé. The group loss for deep metric learning. In *European Conference on Computer Vision*, pages 277–294. Springer, 2020.

[9] Weifeng Ge, Weilin Huang, Dengke Dong, and Matthew R Scott. Deep metric learning with hierarchical triplet loss. In *Proceedings of the European Conference on Computer Vision*, pages 269–285, 2018.

[10] Yixiao Ge, Haibo Wang, Feng Zhu, Rui Zhao, and Hongsheng Li. Self-supervising fine-grained region similarities for large-scale image localization. In *European Conference on Computer Vision*, 2020.

[11] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1735–1742. IEEE, 2006.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[13] Chih-Hui Ho, Pedro Morgado, Amir Persekian, and Nuno Vasconcelos. Pies: Pose invariant embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12377–12386, 2019.

[14] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alexander Shepard, Hartwig Adam, Pietro Perona, and Serge J. Belongie. The inaturalist species classification and detection dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8769–8778, 2018.

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

[16] Hyo Jin Kim and Jan-Michael Frahm. Hierarchy of alternating specialists for scene recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 451–467, 2018.

[17] Sungyeon Kim, Dongwon Kim, Minsu Cho, and Suha Kwak. Proxy anchor loss for deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3238–3247, 2020.

[18] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013.

[19] Elad Levi, Tete Xiao, Xiaolong Wang, and Trevor Darrell. Reducing class collapse in metric learning with easy positive sampling. *arXiv preprint arXiv:2006.05162*, 2020.

[20] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1096–1104, 2016.

[21] Timo Milbich, Karsten Roth, Homanga Bharadhwaj, Samarth Sinha, Yoshua Bengio, Björn Ommer, and Joseph Paul Cohen. Diva: Diverse visual feature aggregation for deep metric learning. In *European Conference on Computer Vision*, pages 590–607. Springer, 2020.

[22] Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 360–368, 2017.

[23] Venkatesh N Murthy, Vivek Singh, Terrence Chen, R Manmatha, and Dorin Comaniciu. Deep decision network for multi-class image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2240–2248, 2016.

[24] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. A metric learning reality check. In *European Conference on Computer Vision*, pages 681–699. Springer, 2020.

[25] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4004–4012, 2016.

[26] Qi Qian, Lei Shang, Baigui Sun, Juhua Hu, Hao Li, and Rong Jin. Softtriple loss: Deep metric learning without triplet sampling. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6450–6458, 2019.

[27] Karsten Roth, Biagio Brattoli, and Bjorn Ommer. Mic: Mining interclass characteristics for improved metric learning. In *Proceedings of the IEEE International Conference on Com-*

puter Vision, pages 8000–8009, 2019.

[28] Sam Roweis, Geoffrey Hinton, and Ruslan Salakhutdinov. Neighbourhood component analysis. *Advances in Neural Information Processing Systems*, 17:513–520, 2004.

[29] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[30] 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.

[31] Artsiom Sanakoyeu, Vadim Tschernezki, Uta Buchler, and Bjorn Ommer. Divide and conquer the embedding space for metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 471–480, 2019.

[32] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[33] Jasper Snoek, Hugo Larochelle, and Ryan Prescott Adams. Practical bayesian optimization of machine learning algorithms. *Advances in Neural Information Processing Systems*, 2012.

[34] Eu Wern Teh, Terrance DeVries, and Graham W Taylor. Proxynca++: Revisiting and revitalizing proxy neighborhood component analysis. In *European Conference on Computer Vision*. Springer, 2020.

[35] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[36] Georg Waltner, Michael Opitz, Horst Possegger, and Horst Bischof. Hibster: Hierarchical boosted deep metric learning for image retrieval. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 599–608. IEEE, 2019.

[37] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5265–5274, 2018.

[38] Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R Scott. Multi-similarity loss with general pair weighting for deep metric learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5022–5030, 2019.

[39] Xun Wang, Haozhi Zhang, Weilin Huang, and Matthew R Scott. Cross-batch memory for embedding learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6388–6397, 2020.

[40] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-ucsd birds 200. 2010.

[41] Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krahenbuhl. Sampling matters in deep embedding learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2840–2848, 2017.

[42] Zhicheng Yan, Hao Zhang, Robinson Piramuthu, Vignesh Jagadeesh, Dennis DeCoste, Wei Di, and Yizhou Yu. Hd-cnn: hierarchical deep convolutional neural networks for large scale visual recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 2740–2748, 2015.

[43] Miin-Shen Yang, Chien-Yo Lai, and Chih-Ying Lin. A robust em clustering algorithm for gaussian mixture models. *Pattern Recognition*, 45(11):3950–3961, 2012.

[44] Xiaohang Zhan, Jiahao Xie, Ziwei Liu, Yew-Soon Ong, and Chen Change Loy. Online deep clustering for unsupervised representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6688–6697, 2020.