ProxyNCA++: Revisiting and Revitalizing Proxy Neighborhood Component Analysis

Eu Wern Teh[1,2], Terrance DeVries[1,2], and Graham W. Taylor[1,2]

[1] University of Guelph, ON, Canada
[2] Vector Institute, ON, Canada
{eteh,terrance,gwtaylor}@uoguelph.ca

Abstract. We consider the problem of distance metric learning (DML), where the task is to learn an effective similarity measure between images. We revisit ProxyNCA and incorporate several enhancements. We find that low temperature scaling is a performance-critical component and explain why it works. Besides, we also discover that Global Max Pooling works better in general when compared to Global Average Pooling. Additionally, our proposed fast moving proxies also addresses small gradient issue of proxies, and this component synergizes well with low temperature scaling and Global Max Pooling. Our enhanced model, called ProxyNCA++, achieves a 22.9 percentage point average improvement of Recall@1 across four different zero-shot retrieval datasets compared to the original ProxyNCA algorithm. Furthermore, we achieve state-of-the-art results on the CUB200, Cars196, Sop, and InShop datasets, achieving Recall@1 scores of 72.2, 90.1, 81.4, and 90.9, respectively.

**Keywords:** Metric Learning; Zero-Shot Learning; Image Retrieval;

1 Introduction

Distance Metric Learning (DML) is the task of learning effective similarity measures between examples. It is often applied to images, and has found numerous applications such as visual products retrieval [16,23,1], person re-identification [37,30], face recognition [22], few-shot learning [28,14], and clustering [11]. In this paper, we focus on DML's application on zero-shot image retrieval [17,33,23,13], where the task is to retrieve images from previously unseen classes.

Proxy-Neighborhood Component Analysis (ProxyNCA) [17] is a proxy-based DML solution that consists of updatable proxies, which are used to represent class distribution. It allows samples to be compared with these proxies instead of one another to reduce computation. After the introduction of ProxyNCA, there are very few works that extend ProxyNCA [35,21], making it less competitive when compared with recent DML solutions [32,13,36].

Our contributions are the following: First, we point out the difference between NCA and ProxyNCA, and propose to use proxy assignment probability which aligns ProxyNCA with NCA [7]. Second, we explain why low temperature scaling works and show that it is a performance-critical component of ProxyNCA.

Third, we explore different global pooling strategies and find out that Global Max Pooling (GMP) outperforms the commonly used Global Average Pooling (GAP), both for ProxyNCA and other methods. Fourth, we suggest using faster moving proxies that compliment well with both GMP and low temperature scaling, which also address the small gradient issue due to L^2-Normalization of proxies. Our enhanced ProxyNCA, which we called ProxyNCA++, has a 22.9 percentage points of improvement over ProxyNCA on average for Recall@1 across four different zero-shot retrieval benchmarks (performance gains are highlighted in Figure 1). In addition, we also achieve state-of-the-art performance on all four benchmark datasets across all categories.

Fig. 1. A summary of the average performance on Recall@1 for all datasets. With our proposed enhancements, we improve upon the original ProxyNCA by 22.9pp, and outperform current state-of-the-art models by 2.0pp on average.
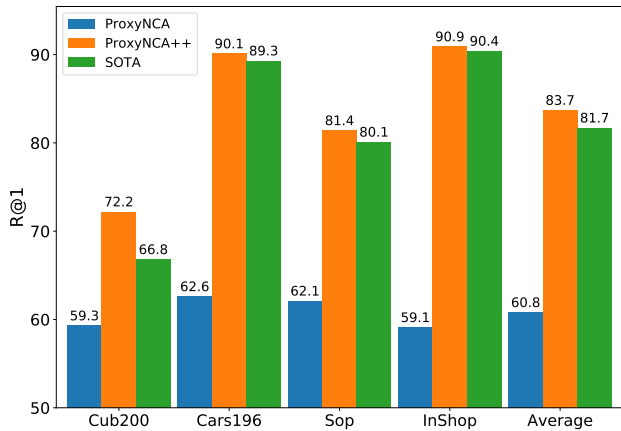
2   Related Work

The core idea of Distance Metric Learning (DML) is to learn an embedding space where similar examples are attracted, and dissimilar examples are repelled. To restrict the scope, we limit our review to methods that consider image data. There is a large body of work in DML, and it can be traced back to the 90s, where Bromley et al. [2] designed a Siamese neural network to verify signatures. Later, DML was used in facial recognition, and dimensionality reduction in the form of a contrastive loss [4,9], where pairs of similar and dissimilar images are selected, and the distance between similar pairs of images is minimized while the distance between dissimilar images is maximized.

Like contrastive loss, which deals with the actual distance between two images, triplet loss optimizes the relative distance between a positive pair (an

anchor image and an image similar to the anchor image) and a negative pair (an anchor image and an image dissimilar to the anchor image) [3]. In addition to contrastive and triplet loss, there is a long line of work which proposes new loss functions, such as angular loss [31], histogram loss [26], margin-based loss [33], and hierarchical triplet loss [6]. Wang et al. [32] categorize this group as paired-based DML.

One weakness of paired-based methods is the sampling process. First, the number of possible pairs grows polynomially with the number of data points, which increases the difficulty of finding an optimal solution. Second, if a pair or triplet of images is sampled randomly, the average distance between two samples is approximately $\sqrt{2}$-away [33]. In other words, a randomly sampled image is highly redundant and provides less information than a carefully chosen one.

In order to overcome the weaknesses of pair-based methods, several works have been proposed in the last few years. Schroff et al. [22] explore a curriculum learning strategy where examples are selected based on the distances of samples to the anchored images. They use a semi-hard negative mining strategy to select negative samples where the distances between negative pairs are at least greater than the positive pairs. However, such a method usually generates very few semi-hard negative samples, and thus requires very large batches (on the order of thousands of samples) in order to be effective. Song et al. [23] propose to utilize all pair-wise samples in a mini-batch to form triplets, where each positive pair compares its distance with all negative pairs. Wu et al. [33] proposed a distance-based sampling strategy, where examples are sampled based on inverse n-dimensional unit sphere distances from anchored samples. Wang et al. [32] propose a mining and weighting scheme, where informative pairs are sampled by measuring positive relative similarity, and then further weighted using self-similarity and negative relative similarity.

Apart from methods dedicated to addressing the weakness of pair-based DML methods, there is another line of work that tackles DML via class distribution estimation. The motivation for this camp of thought is to compare samples to proxies, and in doing so, reduce computation. One method that falls under this line of work is the Magnet Loss [19], in which samples are associated with a cluster centroid, and at each training batch, samples are attracted to cluster centroids of similar classes and repelled by cluster centroids of different classes. Another method in this camp is ProxyNCA [17], where proxies are stored in memory as learnable parameters. During training, each sample is pushed towards its proxy while repelling against all other proxies of different classes. ProxyNCA is discussed in greater detail in Section 3.2.

Similar to ProxyNCA, Zhai et al. [36] design a proxy-based solution that emphasizes on the Cosine distance rather than the Euclidean squared distance. They also use layer norm in their model to improve robustness against poor weight initialization of new parameters and introduces class balanced sampling during training, which improves their retrieval performance. In our work, we also use these enhancements in our architecture.

Recently, a few works in DML have explored ensemble techniques. Opitz et al. [18] train an ensemble DML by reweighting examples using online gradient boosting. The downside of this technique is that it is a sequential process. Xuan et al. [35] address this issue by proposing an ensemble technique where ensemble models are trained separately on randomly combined classes. Sanakoyeu et al. [21] propose a unique divide-and-conquer strategy where the data is divided periodically via clustering based on current combined embedding during training. Each cluster is assigned to a consecutive chunk of the embedding, called learners, and they are randomly updated during training. Apart from ensemble techniques, there is recent work that attempts to improve DML in general. Jacob et al. [13] discover that DML approaches that rely on Global Average Pooling (GAP) potentially suffer from the scattering problem, where features learned with GAP are sensitive to outlier. To tackle this problem, they propose HORDE, which is a high order regularizer for deep embeddings that computes higher-order moments of features.

3   Methods

In this section, we revisit NCA and ProxyNCA and discuss six enhancements that improve the retrieval performance of ProxyNCA. The enhanced version, which we call ProxyNCA++, is shown in Figure 2.

Fig. 2. We show an overview of our architecture, ProxyNCA++, which consists of the original building blocks of ProxyNCA and six enhancements, which are shown in the dashed boxes. ProxyNCA consists of a pre-trained backbone model, a randomly initialized embedding layer, and randomly initialized proxies. The six enhancements in ProxyNCA++ are proxy assignment probability (+prob), low temperature scaling (+scale), class balanced sampling (+cbs), layer norm (+norm), global max pooling (+max) and fast-moving proxies (+fast).
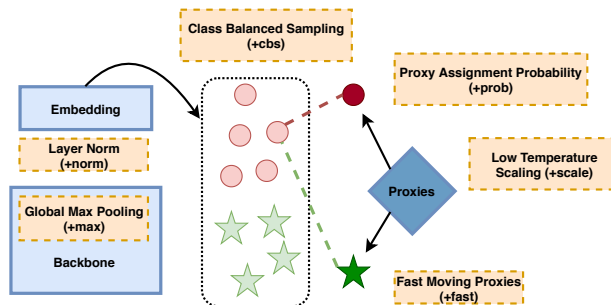
3.1   Neighborhood Component Analysis (NCA)

Neighborhood Component Analysis (NCA) is a DML algorithm that learns a Mahalanobis distance for k-nearest neighbors (KNN). Given two points, x_i and x_j, Goldberg et al. [7] define p_{ij} as the assignment probability of x_i to x_j:

$$p_{ij} = \frac{-d(x_i, x_j)}{\sum_{k \notin i} -d(x_i, x_k)} \tag{1}$$

where $d(x_i, x_k)$ is Euclidean squared distance computed on some learned embedding. In the original work, it was parameterized as a linear mapping, but nowadays, the method is often used with nonlinear mappings such as feedforward or convolutional neural networks. Informally, p_{ij} is the probability that points i and j are said to be "neighbors".

The goal of NCA is to maximize the probability that points assigned to the same class are neighbors, which, by normalization, minimizes the probability that points in different classes are neighbors:

$$L_{\mathrm{NCA}} = -\log\left(\frac{\sum_{j \in C_i} \exp(-d(x_i, x_j))}{\sum_{k \notin C_i} \exp(-d(x_i, x_k))}\right). \tag{2}$$

Unfortunately, the computation of NCA loss grows polynomially with the number of samples in the dataset. To speed up computation, Goldberg et al. use random sampling and optimize the NCA loss with respect to the small batches of samples.

3.2   ProxyNCA

ProxyNCA is a DML method which performs metric learning in the space of class distributions. It is motivated by NCA, and it attempts to address the computational weakness of NCA by using proxies. In ProxyNCA, *proxies* are stored as learnable parameters to faithfully represent classes by prototypes in an embedding space. During training, instead of comparing samples with one another in a given batch, which is quadratic in computation with respect to the batch size, ProxyNCA compares samples against proxies, where the objective aims to attract samples to their proxies and repel them from all other proxies.

Let C_i denote a set of points that belong to the same class, $f(a)$ be a proxy function that returns a corresponding class proxy, and $||a||_2$ be the L^2-Norm of vector a. For each sample x_i, we minimize the distance $d(x_i, f(x_i))$ between the sample, x_i and its own proxy, $f(x_i)$ and maximize the distance $d(x_i, f(z))$ of that sample with respect to all other proxies Z, where $f(z) \in Z$ and $z \notin C_i$.

$$L_{\mathrm{ProxyNCA}} = -\log\left(\frac{\exp\left(-d\left(\frac{x_i}{||x_i||_2}, \frac{f(x_i)}{||f(x_i)||_2}\right)\right)}{\sum_{f(z) \in Z} \exp\left(-d\left(\frac{x_i}{||x_i||_2}, \frac{f(z)}{||f(z)||_2}\right)\right)}\right). \tag{3}$$

3.3   Aligning with NCA by optimizing proxy assignment probability

Using the same motivation as NCA (Equation 1), we propose to optimize the proxy assignment probability, P_i. Let A denote the set of all proxies. For each x_i, we aim to maximize P_i.

$$P_i = \frac{\exp\left(-d(\frac{x_i}{||x_i||_2}, \frac{f(x_i)}{||f(x_i)||_2})\right)}{\sum_{f(a) \in A} \exp\left(-d(\frac{x_i}{||x_i||_2}, \frac{f(a)}{||f(a)||_2})\right)} \tag{4}$$

$$L_{\text{ProxyNCA++}} = -\log(P_i) \tag{5}$$

Since P_i is a probability score that must sum to one, maximizing P_i for a proxy also means there is less chance for x_i to be assigned to other proxies. In addition, maximizing P_i also preserves the original ProxyNCA properties where x_i is attracted toward its own proxy $f(x_i)$ while repelling proxies of other classes, Z. It is important to note that in ProxyNCA, we maximize the distant ratio between $-d(x_i, y_j)$ and $\sum_{f(z) \in Z} -d(x_i, f(z))$, while in ProxyNCA++, we maximize the proxy assignment probability, P_i, a subtle but important distinction. Table 8 shows the effect of proxy assignment probability to ProxyNCA and its enhancements.

3.4   About Temperature Scaling

Temperature scaling is introduced in [12], where Hinton et al. use a high temperature $(T > 1)$ to create a softer probability distribution over classes for knowledge distillation purposes. Given a logit y_i and a temperature variable T, a temperature scaling is defined as $q_i = \frac{\exp(y_i/T)}{\sum_j \exp(y_j/T)}$. By incorporating temperature scaling to the loss function of ProxyNCA++ in Equation 4, the new loss function has the following form:

$$L_{\text{ProxyNCA++}} = -\log\left(\frac{\exp\left(-d(\frac{x_i}{||x_i||_2}, \frac{f(x_i)}{||f(x_i)||_2}) * \frac{1}{T}\right)}{\sum_{f(a) \in A} \exp\left(-d(\frac{x_i}{||x_i||_2}, \frac{f(a)}{||f(a)||_2}) * \frac{1}{T}\right)}\right) \tag{6}$$

When $T = 1$, we have a regular Softmax function. As T gets larger, the output of the softmax function will approach a uniform distribution. On the other hand, as T gets smaller, it leads to a peakier probability distribution. Low temperature scaling $(T < 1)$ is used in [34] and [36]. In this work, we attempt to explain why low-temperature scaling works by visualizing its effect on synthetic data. In Figure 3, as T gets smaller, the decision boundary is getting more refined and can classify the samples better. In other words, as T becomes smaller, the model can overfit to the problem better and hence generating better decision boundaries.

In Figure 4 (a), we show a plot of R@1 score with respect to temperature scale on the CUB200 dataset. The highest test average R@1 happens at $T =$

$\frac{1}{9}$. Lowering T beyond this point will allow the model to overfit more to the training set and to make it less generalizable. Hence, we see a drop in test performance. Table 9 shows the effect of low temperature scaling to ProxyNCA and its enhancements.

Fig. 3. The effect of temperature scaling on the decision boundary of a Softmax Classifier trained on the two moons synthetic dataset
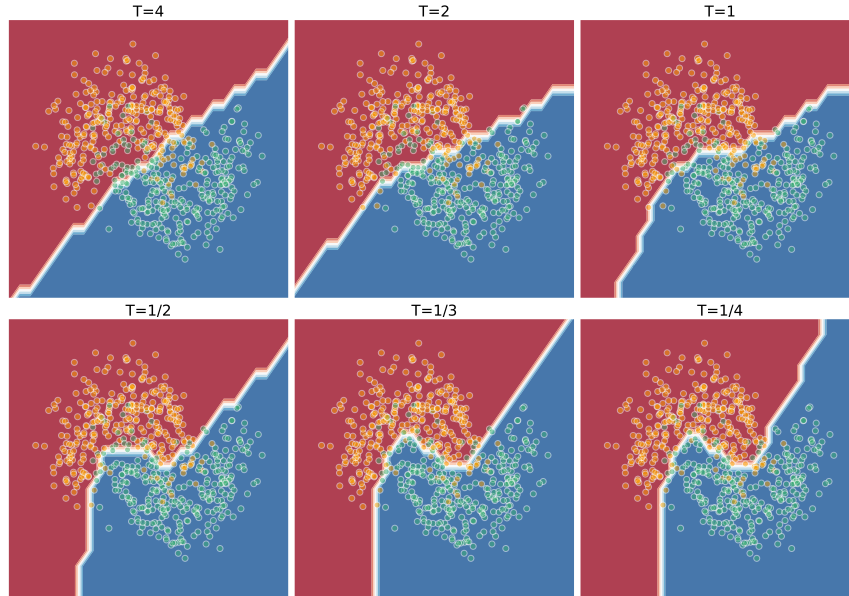
3.5 About Global Pooling

In DML, the de facto global pooling operation used by the community is Global Average Pooling (GAP). In this paper, we investigate the effect of global pooling of spatial features on zero-shot image retrieval. We propose the use of Global K-Max Pooling [5] to interpolate between GAP and Global Max Pooling (GMP). Given a convolution feature map of $M \times M$ dimension with E channels, $g \in \mathbb{R}^{M \times M \times E}$ and a binary variable, $h_i \in \{0, 1\}$, Global K-Max Pooling is defined as:

$$\text{Global } k\text{-Max}(g_\epsilon) = \max_h \frac{1}{k} \sum_{i=1}^{M^2} h_i \cdot g_\epsilon \text{ , s.t.} \sum_{i=1}^{M^2} h_i = k, \forall \epsilon \in E \qquad (7)$$

When $k = 1$, we have GMP, and when $k = M^2$, we have GAP. Figure 4 (b) is a plot of Recall@1 with different k value of Global K-Max Pooling on

Fig. 4. We show three plots of R@1 with different (a) temperature scales , (b) k values for K-Max Pooling and (c) proxy learning rates on on CUB200 [29]. The shaded areas represent one standard deviation of uncertainty.
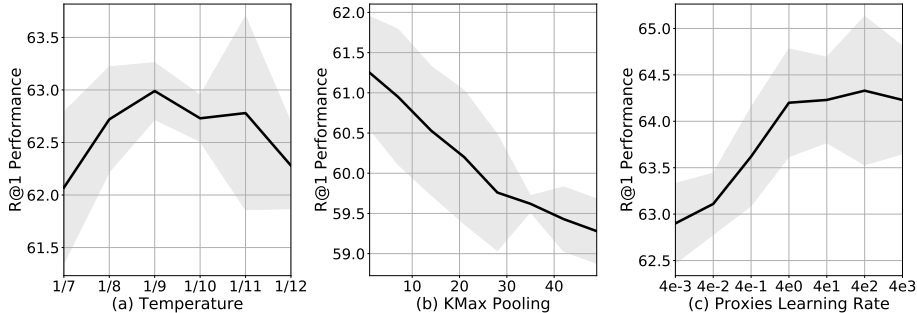
the CUB200 dataset. There is a negative correlation of 0.98 between k and Recall@1 performance, which shows that a lower k value results in better retrieval performance.

3.6   About Fast moving proxies

In ProxyNCA, the proxies, the embedding layer, and the backbone model all share the same learning rate. We hypothesize that the proxies should be moving faster than the embedding space in order to represent the class distribution better. However, in our experiments, we discovered that the gradient of proxies is smaller than the gradient of the embedding layer and backbone model by three orders of magnitude, and this is caused by the L^2-Normalization of proxies. To mitigate this problem, we use a higher learning rate for the proxies.

From our ablation studies in Table 9, we observe that fast moving proxies synergize better with low temperature scaling and Global Max Pooling. We can see a 1.4pp boost in R@1 if we combine fast proxies and low temperature scaling. There is also a 2.1pp boost in the retrieval performance if we combine fast proxies, low temperature scaling, and Global Max Pooling. Figure 4 (c) is a plot of Recall@1 with different proxy learning rates on CUB200.

3.7   Layer Norm (Norm) and Class Balanced Sampling (CBS)

The use of layer normalization [27] without affine parameters is explored by Zhai et al. [36]. Based on our experiments, we also find that this enhancement helps to boost performance. Besides, we also use a class balanced sampling strategy in our experiments, where we have more than one instance per class in each training batch. To be specific, for every batch of size N_b, we only sample N_c classes from which we then randomly select $\lfloor N_b/N_c \rfloor$ examples. This sampling strategy commonly appears in pair-based DML approaches [23,32,33] as a baseline and Zhai et al. is the first paper that uses it in a proxy-based DML method.

4   Experiments

We train and evaluate our model on four zero-shot image retrieval datasets: the Caltech-UCSD Birds dataset [29] (CUB200), the Stanford Cars dataset [15] (Cars196), the Stanford Online Products dataset [23] (Sop), and the In Shop Clothing Retrieval dataset [16] (InShop). The composition in terms of number of images and classes of each dataset is summarized in Table 1.

4.1   Experimental Setup

For each dataset, we use the first half of the original training set as our training set and the second half of the original training set as our validation set. In all of our experiments, we use a two-stage training process. We first train our models on the training set and then use the validation set to perform hyper-parameter tuning (e.g., selecting the best epoch for early stopping, learning rate, etc.). Next, we train our models with the fine-tuned hyper-parameters on the combined training and validation sets (i.e., the complete original training set).

Table 1. We show the composition of all four zero-shot image retrieval datasets considered in this work. In addition, we also report the learning rates, the batch size, and cbs (class balanced sampling) instances for each dataset during training. The number of classes for the Sop and InShop datasets is large when compared to CUB200 and Cars196 dataset. However, the number of instances per class is very low for the Sop and InShop datasets. In general, ProxyNCA does not require a large batch size when compared to pairs-based DML methods. To illustrate this, we also show the batch sizes used in [32], which is current state-of-the-art among pairs-based methods. Their technique requires a batch size which is several times larger than ProxyNCA++.

	images	classes	avg	batch size (ours)	batch size (MS [32])	Base lr	Proxy lr	cbs
CUB200	11,788	200	58	32	80	4e-3	4e2	4
Cars196	16,185	196	82	32	-	4e-3	4e2	4
Sop	120,053	22,634	5	192	1000	2.4e-2	2.4e2	3
InShop	52,712	11,967	4	192	-	2.4e-2	2.4e3	3

We use the same learning rate for both stages of training. We also set the number of proxies to be the same as the number of classes in the training set. For our experiments with fast proxies, we use a different learning rate for proxies (see Table 1 for details). We also use a temperature value of $\frac{1}{9}$ across all datasets.

In the first stage of training, we use the "reduce on loss plateau decay" annealing [8] to control the learning rate of our model based on the recall performance (R@1) on the validation set. We set the patience value to four epochs in our experiments. We record the epochs where the learning rate is reduced and also save the best epochs for early stopping on the second stage of training.

In all of our experiments, we leverage the commonly used ImageNet [20] pre-trained Resnet50 [10] model as our backbone (see Table 2 for commonly

used backbone architectures). Features are extracted after the final convolutional block of the model and are reduced to a spatial dimension of 1×1 using a global pooling operation. This procedure results in a 2048 dimensional vector, which is fed into a final embedding layer. In addition, we also experiment with various embedding sizes. We observe a gain in performance as we increase the size of the embedding. It is important to note that not all DML techniques yield better performance as embedding size increases. For some techniques such as [32,23], a larger embedding size hurts performance.

Table 2. Commonly used backbone architectures for zero-shot image retrieval, with associated ImageNet Top-1 Error % for each architecture

Architecture	Abbreviation	Top-1 Error (%)
Resnet18 [10]	R18	30.24
GoogleNet [25]	I1	30.22
Resnet50 [10]	R50	23.85
InceptionV3 [24]	I3	22.55

During training, we scale the original images to a random aspect ratio (0.75 to 1.33) before applying a crop of random size (0.08 to 1.0 of the scaled image). After cropping, we resize the images to 256×256. We also perform random horizontal flipping for additional augmentation. During testing, we resize the images to 288×288 and perform a center crop of size 256×256.

4.2   Evaluation

We evaluate retrieval performance based on two evaluation metrics: (a) Recall@K (R@K) and (b) Normalized Mutual Information, $\mathrm{NMI}(\Omega, \mathbb{C}) = \frac{2*I(\Omega,\mathbb{C})}{H(\Omega)+H(\mathbb{C})}$, where Ω represents ground truth label, $\mathbb{C}$ represents the set of clusters computed by K-means, I stands for mutual information and H stands for entropy. The purpose of NMI is to measure the purity of the cluster on unseen data.

Using the same evaluation protocols detailed in [17,32,13,16], we evaluate our model using unseen classes on four datasets. The InShop dataset [16] is slightly different than all three other datasets. There are three groupings of data: training set, query set, and gallery set. The query and gallery set have the same classes, and these classes do not overlap with the training set. Evaluation is done based on retrieval performance on the gallery set.

Tables 3, 4, 5, and 6 show the results of our experiments [1]. For each dataset, we report the results of our method, averaged over five runs. We also report the standard deviation of our results to account for uncertainty. Additionally, we also show the results of ProxyNCA++ trained with smaller embedding sizes (512,

---

[1] For additional experiments on different crop sizes, please refer to the corresponding supplementary materials

1024). Our ProxyNCA++ model outperforms ProxyNCA and all other state-of-the-art methods in all categories across all four datasets. Note, our model trained with a 512-dimensional embedding also outperform all other methods in the same embedding space except for The InShop dataset [16], where we tie in the R@1 category.

Table 3. Recall@k for k = 1,2,4,8 and NMI on CUB200-2011 [29]

R@k	1	2	4	8	NMI	Arch	Emb
ProxyNCA[17]	49.2	61.9	67.9	72.4	59.5	I1	128
Margin[33]	63.6	74.4	83.1	90.0	69.0	R50	128
MS [32]	65.7	77.0	86.3	91.2	-	I3	512
HORDE [13]	66.8	77.4	85.1	91.0	-	I3	512
NormSoftMax [36]	61.3	73.9	83.5	90.0	-	R50	512
NormSoftMax [36]	65.3	76.7	85.4	91.8	-	R50	2048
ProxyNCA	59.3±0.4	71.2±0.3	80.7±0.2	88.1±0.3	63.3±0.5	R50	2048
ProxyNCA++	69.0±0.8	79.8±0.7	87.3±0.7	92.7±0.4	73.9±0.5	R50	512
ProxyNCA++	70.2±1.6	80.7±1.4	88.0±0.9	93.0±0.4	74.2±1.0	R50	1024
ProxyNCA++ (-max, -fast)	69.1±0.5	79.6±0.4	87.3±0.3	92.7±0.2	73.3±0.7	R50	2048
ProxyNCA++	**72.2±0.8**	**82.0±0.6**	**89.2±0.6**	**93.5±0.4**	**75.8±0.8**	R50	2048

Table 4. Recall@k for k = 1,2,4,8 and NMI on CARS196 [15]

R@k	1	2	4	8	NMI	Arch	Emb
ProxyNCA [17]	73.2	82.4	86.4	88.7	64.9	I1	128
Margin [33]	79.6	86.5	91.9	95.1	69.1	R50	128
MS [32]	84.1	90.4	94.0	96.1	-	I3	512
HORDE [13]	86.2	91.9	95.1	97.2	-	I3	512
NormSoftMax [36]	84.2	90.4	94.4	96.9	-	R50	512
NormSoftMax [36]	89.3	94.1	96.4	98.0	-	R50	2048
ProxyNCA	62.6±9.1	73.6±8.6	82.2±6.9	88.9±4.8	53.8±7.0	R50	2048
ProxyNCA++	86.5±0.4	92.5±0.3	95.7±0.2	97.7±0.1	73.8±1.0	R50	512
ProxyNCA++	87.6±0.3	93.1±0.1	96.1±0.2	97.9±0.1	75.7±0.3	R50	1024
ProxyNCA++ (-max, -fast)	87.9±0.2	93.2±0.2	96.1±0.2	97.9±0.1	76.0±0.5	R50	2048
ProxyNCA++	**90.1±0.2**	**94.5±0.2**	**97.0±0.2**	**98.4±0.1**	**76.6±0.7**	R50	2048

4.3 Ablation Study

In Table 7, we perform an ablation study on the performance of our proposed methods using the CUB200 dataset. The removal of the low temperature scaling component gives the most significant drop in R@1 performance (-10.8pt).

Table 5. Recall@k for k = 1,10,100,1000 and NMI on Stanford Online Products [23].

R@k	1	10	100	1000	Arch	Emb
ProxyNCA [17]	73.7	-	-	-	I1	128
Margin [33]	72.7	86.2	93.8	98.0	R50	128
MS [32]	78.2	90.5	96.0	98.7	I3	512
HORDE [13]	80.1	91.3	96.2	98.7	I3	512
NormSoftMax [36]	78.2	90.6	96.2	-	R50	512
NormSoftMax [36]	79.5	91.5	96.7	-	R50	2048
ProxyNCA	62.1±0.4	76.2±0.4	86.4±0.2	93.6±0.3	R50	2048
ProxyNCA++	80.7±0.5	92.0±0.3	96.7±0.1	98.9±0.0	R50	512
ProxyNCA++	80.7±0.4	92.0±0.2	96.7±0.1	98.9±0.0	R50	1024
ProxyNCA++(-max, -fast)	72.1±0.2	85.4±0.1	93.0±0.1	96.7±0.2	R50	2048
ProxyNCA++	**81.4±0.1**	**92.4±0.1**	**96.9±0.0**	**99.0±0.0**	R50	2048

Table 6. Recall@k for k = 1,10,20,30,40 on the In-Shop Clothing Retrieval dataset [23]

R@k	1	10	20	30	40	Arch	Emb
MS [32]	89.7	97.9	98.5	98.8	99.1	I3	512
HORDE [13]	90.4	97.8	98.4	98.7	98.9	I3	512
NormSoftMax [36]	88.6	97.5	98.4	98.8	-	R50	512
NormSoftMax [36]	89.4	97.8	98.7	99.0	-	R50	2048
ProxyNCA	59.1±0.7	80.6±0.6	84.7±0.3	86.7±0.4	88.1±0.5	R50	2048
ProxyNCA++	90.4±0.2	98.1±0.1	98.8±0.0	99.0±0.1	99.2±0.0	R50	512
ProxyNCA++	90.4±0.4	98.1±0.1	98.8±0.1	99.1±0.1	99.2±0.1	R50	1024
ProxyNCA++ (-max, -fast)	82.5±0.3	93.5±0.1	95.4±0.2	96.3±0.0	96.8±0.0	R50	2048
ProxyNCA++	**90.9±0.3**	**98.2±0.0**	**98.9±0.0**	**99.1±0.0**	**99.4±0.0**	R50	2048

This is followed by Global Max Pooling (-3.2pt), Layer Normalization (-2.6pt), Class Balanced Sampling (-2.6pt), Fast proxies (-1.9pt) and Proxy Assignment Probability (-1.1pt).

We compare the effect of the Global Max Pooling (GMP) and the Global Average Pooling (GAP) on other metric learning methodologies [22,33,32,13] in Table 11 on CUB200 dataset. The performance of all other models improves when GAP is replaced with GMP, with the exception of HORDE [13]. In HORDE, Jacob et al. [13] include both the pooling features as well as the higher-order moment features in the loss calculation. We speculate that since this method is designed to reduce the effect of outliers, summing max-pooled features canceled out the effect of higher-order moment features, which may have lead to sub-optimal performance.

5   Conclusion

We revisit ProxyNCA and incorporate several enhancements. We find that low temperature scaling is a performance-critical component and explain why it works. Besides, we also discover that Global Max Pooling works better in general

Table 7. An ablation study of ProxyNCA++ and its enhancements on CUB200 [29].

R@k	1	2	4	8	NMI
ProxyNCA++ (Emb: 2048)	72.2±0.8	82.0±0.6	89.2±0.6	93.5±0.4	75.8±0.8
-scale	61.4±0.4	72.4±0.5	81.5±0.3	88.4±0.5	64.8±0.4
-max	69.0±0.6	80.3±0.5	88.1±0.4	93.1±0.1	74.3±0.4
-norm	69.6±0.3	80.5±0.5	88.0±0.2	93.0±0.2	75.2±0.4
-cbs	69.6±0.6	80.1±0.3	87.7±0.3	92.8±0.2	73.4±0.3
-fast	70.3±0.9	80.6±0.4	87.7±0.5	92.5±0.3	73.5±0.9
-prob	71.1±0.7	81.1±0.3	87.9±0.3	92.6±0.3	73.4±0.8

when compared to Global Average Pooling. Additionally, our proposed fast moving proxies also addresses small gradient issue of proxies, and this component synergizes well with low temperature scaling and Global Average pooling. The new and improved ProxyNCA, which we call ProxyNCA++, outperforms the original ProxyNCA by 22.9 percentage points on average across four zero-shot image retrieval datasets for Recall@1. In addition, we also achieve state-of-art results on all four benchmark datasets for all categories.

Table 8. An ablation study of the effect of Proxy Assignment Probability (+prob) to ProxyNCA and its enhancements on CUB200 [29].

R@1	without prob	with prob
ProxyNCA (Emb: 2048)	59.3 ± 0.4	59.0 ± 0.4
+scale	62.9 ± 0.4	63.4 ± 0.6
+scale +norm	65.3 ± 0.7	65.7 ± 0.8
+scale +max	65.1 ± 0.3	66.2 ± 0.3
+scale +norm +cbs	67.2 ± 0.8	69.1 ± 0.5
+scale +norm +cbs +max	68.8 ± 0.7	70.3 ± 0.9
+scale +norm +cbs +max +fast	71.1 ± 0.7	72.2 ± 0.8

Table 9. An ablation study of the effect of low temperature scaling to ProxyNCA and its enhancements on CUB200 [29]. Without low temperature scaling, three out of six enhancements (in red) get detrimental results when they are applied to ProxyNCA.

R@1	without scale	with scale
ProxyNCA (Emb: 2048)	59.3 ± 0.4	62.9 ± 0.4
+cbs	54.8 ± 6.2	64.0 ± 0.4
+prob	59.0 ± 0.4	63.4 ± 0.6
+norm	60.2 ± 0.6	65.3 ± 0.7
+max	61.3 ± 0.7	65.1 ± 0.3
+fast	56.3 ± 0.8	64.3 ± 0.8
+max +fast	60.3 ± 0.5	67.2 ± 0.5
+norm +prob +cbs	60.4 ± 0.7	69.1 ± 0.5
+norm +prob +cbs +max	61.2 ± 0.7	70.3 ± 0.9
+norm +prob +cbs +max +fast	61.4 ± 0.4	72.2 ± 0.8

Table 10. An ablation study of ProxyNCA the effect of Global Max Pooling to ProxyNCA and its enhancements on CUB200 [29]. We can see a 2.1pp improvement on average after replacing GAP with GMP.

R@1	Global Average Pooling	Global Max Pooling
ProxyNCA (Emb: 2048)	59.3 ± 0.4	61.3 ± 0.7
+cbs	54.8 ± 6.2	55.5 ± 6.2
+prob	59.0 ± 0.4	61.2 ± 0.7
+norm	60.2 ± 0.6	60.9 ± 0.9
+scale	62.9 ± 0.4	65.1 ± 0.3
+fast	56.3 ± 0.8	60.3 ± 0.5
+scale +fast	64.3 ± 0.8	67.2 ± 0.5
+norm +prob +cbs	60.4 ± 0.7	61.2 ± 0.7
+norm +prob +cbs +fast	56.2 ± 0.9	61.4 ± 0.4
+norm +prob +cbs +scale	69.1 ± 0.5	70.3 ± 0.9
+norm +prob +cbs +scale +fast	69.0 ± 0.6	72.2 ± 0.8

Table 11. Comparing the effect of Global Max Pooling and Global Average Pooling on the CUB200 dataset for a variety of methods.

Method	Pool	R@1	Arch	Emb
WithoutTraining	avg	45.0	R50	2048
	max	**53.1**	R50	2048
Margin [33]	avg	63.3	R50	128
	max	**64.3**	R50	128
Triplet-Semihard sampling [22]	avg	60.5	R50	128
	max	**61.6**	R50	128
MS [32]	avg	64.9	R50	512
	max	**68.5**	R50	512
MS [32]	avg	65.1	I3	512
	max	**66.1**	I3	512
Horde (Contrastive Loss) [13]	avg	**65.1**	I3	512
	max	63.1	I3	512

References

1. Sean Bell and Kavita Bala. Learning visual similarity for product design with convolutional neural networks. *ACM Trans. Graph.*, 34(4):98:1–98:10, July 2015. 1

2. Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. In *Proceedings of the 6th International Conference on Neural Information Processing Systems*, NIPS'93, pages 737–744, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc. 2

3. Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. Large scale online learning of image similarity through ranking. *J. Mach. Learn. Res.*, 11:1109–1135, Mar. 2010. 3

4. Sumit Chopra, Raia Hadsell, Yann LeCun, et al. Learning a similarity metric discriminatively, with application to face verification. 2

5. Thibaut Durand, Nicolas Thome, and Matthieu Cord. Weldon: Weakly supervised learning of deep convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4743–4752, 2016. 7

6. Weifeng Ge. Deep metric learning with hierarchical triplet loss. In *The European Conference on Computer Vision (ECCV)*, September 2018. 3

7. Jacob Goldberger, Geoffrey E Hinton, Sam T Roweis, and Ruslan R Salakhutdinov. Neighbourhood components analysis. In *Advances in neural information processing systems*, pages 513–520, 2005. 1, 5

8. Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org. 9

9. Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006. 2

10. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 9, 10

11. J. R. Hershey, Z. Chen, J. Le Roux, and S. Watanabe. Deep clustering: Discriminative embeddings for segmentation and separation. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 31–35, March 2016. 1

12. Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 6

13. Pierre Jacob, David Picard, Aymeric Histace, and Edouard Klein. Metric learning with horde: High-order regularizer for deep embeddings. *arXiv preprint arXiv:1908.02735*, 2019. 1, 4, 10, 11, 12, 14

14. Gregory Koch. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, 2015. 1

15. 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. 9, 11

16. 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. 1, 9, 10, 11

17. 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. 1, 3, 10, 11, 12

18. Michael Opitz, Georg Waltner, Horst Possegger, and Horst Bischof. Bier - boosting independent embeddings robustly. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 4

19. Oren Rippel, Manohar Paluri, Piotr Dollar, and Lubomir Bourdev. Metric learning with adaptive density discrimination. *arXiv preprint arXiv:1511.05939*, 2015. 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. 9

21. Artsiom Sanakoyeu, Vadim Tschernezki, Uta Buchler, and Bjorn Ommer. Divide and conquer the embedding space for metric learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 471–480, 2019. 1, 4

22. Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 1, 3, 12, 14

23. Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 3, 8, 9, 10, 12

24. Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 10

25. C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, June 2015. 10

26. Evgeniya Ustinova and Victor Lempitsky. Learning deep embeddings with histogram loss. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4170–4178. Curran Associates, Inc., 2016. 3

27. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 8

28. Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pages 3637–3645, USA, 2016. Curran Associates Inc. 1

29. Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 8, 9, 11, 13, 14

30. Guanshuo Wang, Yufeng Yuan, Xiong Chen, Jiwei Li, and Xi Zhou. Learning discriminative features with multiple granularities for person re-identification. In *Proceedings of the 26th ACM International Conference on Multimedia*, MM '18, pages 274–282, New York, NY, USA, 2018. ACM. 1

31. Jian Wang, Feng Zhou, Shilei Wen, Xiao Liu, and Yuanqing Lin. Deep metric learning with angular loss. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 3

32. 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. 1, 3, 8, 9, 10, 11, 12, 14
33. 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. 1, 3, 8, 11, 12, 14
34. Zhirong Wu, Alexei A Efros, and Stella X Yu. Improving generalization via scalable neighborhood component analysis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 685–701, 2018. 6
35. Hong Xuan, Richard Souvenir, and Robert Pless. Deep randomized ensembles for metric learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 723–734, 2018. 1, 4
36. Andrew Zhai, Hao-Yu Wu, and US San Francisco. Classification is a strong baseline for deep metric learning. 2019. 1, 3, 6, 8, 11, 12
37. Feng Zheng, Cheng Deng, Xing Sun, Xinyang Jiang, Xiaowei Guo, Zongqiao Yu, Feiyue Huang, and Rongrong Ji. Pyramidal person re-identification via multi-loss dynamic training. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1