Classification is a Strong Baseline for Deep Metric Learning

Andrew Zhai*
andrew@pinterest.com
Hao-Yu Wu*
rexwu@pinterest.com
Pinterest, Inc.
San Francisco, US

Abstract

Deep metric learning aims to learn a function mapping image pixels to embedding feature vectors that model the similarity between images. Two major applications of metric learning are content-based image retrieval and face verification. For the retrieval tasks, the majority of current state-of-the-art (SOTA) approaches are triplet-based non-parametric training. For the face verification tasks, however, recent SOTA approaches have adopted classification-based parametric training. In this paper, we look into the effectiveness of classification based approaches on image retrieval datasets. We evaluate on several standard retrieval datasets such as CAR-196, CUB-200-2011, Stanford Online Product, and In-Shop datasets for image retrieval and clustering, and establish that our classification-based approach is competitive across different feature dimensions and base feature networks. We further provide insights into the performance effects of subsampling classes for scalable classification-based training, and the effects of binarization, enabling efficient storage and computation for practical applications.

1 Introduction

Metric learning, also known as learning image embeddings, is a core problem of a variety of applications including face recognition [13, 16, 22, 23], fine-grained retrieval [17, 18, 27], clustering [31], and visual search [7, 11, 32, 33]. The goal of metric learning is that the learned embedding generalize well to novel instances during test time, an open-set setup in machine learning. This goal aligns well with practical applications in which the deployed machine learning system is required to handle large amount of unseen data.

Standard deep neural network metric learning methods train image embeddings through the local relationships between images in the form of *pairs* [1, 2] or *triplets* [6, 16]. A core challenge with metric learning is sampling informative samples for training. As described in [5, 16, 27], negatives that are too hard can destabilize training, while negatives that are too easy result in triplets that have near zero loss leading to slow convergence.

Recent methods such as [5, 17, 18, 27] focus on addressing this sampling problem, many of which utilize the relationships of all images within the *batch* to form informative triplets. These methods typically require a large batch size so that informative triplets can be selected

*These two authors contributed equally to this work

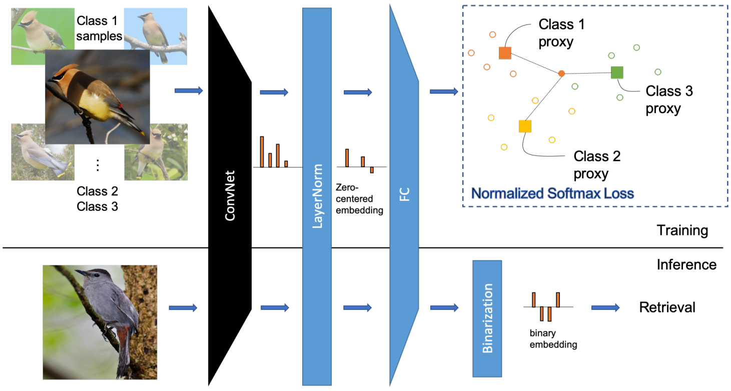

Figure 1: Architecture overview for training high dimensional binary embedding

within the batch. In practice, batch size is constrained by the hardware memory size. Therefore, as the dataset size grows, one still faces the challenge of the diminishing probability of a randomly sampled batch containing any informative triplet.

To alleviate the challenge of sampling informative triplets, another school of deep metric learning approaches propose to use classification-based training [13, 14, 22, 23, 29]. In contrast to triplet-based approaches, the classification-based approach can be viewed as approximating each class using a proxy [14], and uses all proxies to provide global context for each training iteration. Though classification-based metric learning simplify training by removing sampling, they have scalability limitations as in the *extreme classification* [18] problem, where it is considered impractical to extend to much larger number of classes during training.

While generic deep metric learning approaches have been pushing SOTA through better negative sampling and ensembles of triplet-based approaches, face verification has in parallel seen SOTA results through classification-based loss functions. We set out to investigate if classification-based training can perform similarly well for general image retrieval tasks.

Our major contributions are as follows: 1) we establish that classification is a strong baseline for deep metric learning across different datasets, base feature networks and embedding dimensions, 2) we provide insights into the performance effects of binarization and subsampling classes for scalable *extreme classification*-based training 3) we propose a classification-based approach to learn high-dimensional binary embeddings that achieve state-of-the-art retrieval performance with the same memory footprint as 64 dimensional float embeddings across the suite of retrieval tasks.

2 Related Works

Metric Learning Losses Metric learning approaches aim to learn a good embedding space such that the similarity between samples are preserved as distance between embedding vectors of the samples. The metric learning losses, such as contrastive loss[2] and triplet loss[6], are formulated to minimize intra-class distances and maximize inter-class distances. Recent approaches in metric learning design the loss function to consider the relationships of all the samples within the training batch[9, 17, 18, 21, 27], and achieve state-of-the-art performance in image retrieval datasets[12, 18, 25].

Training Sampling Sampling informative training samples plays an important role in metric learning as also suggested in [16, 27]. Semi-hard sampling in conjunction with triplet-loss [16] has been widely adopted for many tasks. Distanced-weighted sampling [27] suggests that with a balanced mix of difficulties during training, the image retrieval performance can be further improved. Hierarchical Triplet Loss [24] proposed that by merging similar classes dynamically during training into a hierarchical tree, more informative samples can be drawn from such a structure and the loss also provides global context for training.

Classification Losses for Metric Learning Classification losses are widely adopted in face verification applications [13, 22, 23] and achieve state-of-the-art performance. The theoretical link between classification and metric learning is shown in [14], and image retrieval tasks have some success adopting classification loss [14, 29]. Though classification-based training alleviates the need for carefully choosing sampling method, the parametric nature may cause issue in fine-grained open-set recognition setting [28].

Ensembling Ensembling embeddings has been the most recent focus to further improve image retrieval performance. The ensembled embeddings are trained via boosting [15], attending diverse spatial locations [26], or partitioning the training classes [29]. However, such ensembled embeddings trade off image retrieval performance with higher dimensions. They also introduce additional hyperparameters into the system.

3 Method

We study recent model architectures and triplet-based and classification losses in Section 4. Beyond standard classification training [4], we describe below techniques we used to achieve SOTA on the retrieval tasks including L2 normalization of embedding to optimize for cosine similarity, Layer Normalization of final pooled feature layer, and class balanced sampling of minibatch. An overview of our approach is shown in Figure 1.

3.1 Normalized Softmax Loss

When training the classification network for metric learning, we remove the bias term in the last linear layer and add an L2 normalization module to the inputs and weights before softmax loss to optimize for cosine similarity. Temperature scaling is used for exaggerating the difference among classes and boosting the gradients as also used in [13, 23, 28]. We follow the same derivation and notations as in [14]. For embedding x of input image with class label y, the loss with temperature σ can be expressed with the weight of class y p_y among all possible classes set Z:

$$L_{\text{norm}} = -\log\left(\frac{\exp(x^T p_y/\sigma))}{\sum_{z\in Z}\exp(x^T p_z/\sigma))}\right) \tag{1}$$

Normalized softmax loss fits into the proxy paradigm when we view the class weight as proxy and choose the distance metric as cosine distance function. A more general form of classification loss, Large Margin Cosine Loss (LMCL), has been proposed for face verification [23] with an additional margin hyperparameter. We also evaluate the effect of LMCL in conjunction with our proposed techniques on image retrieval tasks in Section 4.6.

3.2 Layer Normalization

The layer normalization without affine parameters[10] is added immediately after the final pooling layer of the feature model (e.g. GoogleNet [20]'s pool5 layer) to normalize the feature dimension of our embeddings to have a distribution of values centered at zero. This allows us to easily binarize embeddings via thresholding at zero. We also show empirically through ablation experiments in Section 4.4.1 that layer normalization helps the network better initialize new parameters and reach better optima.

3.3 Class Balanced Sampling

Based on the proxy explanation of using classification loss for metric learning, the loss is bounded by the worst approximated examples within the class [14]. A simple way to alleviate this is by including multiple examples per class when constructing the training mini batch. For each training batch, we first sample C classes, and sample S samples within each class. The effect of class balanced sampling is studied through ablation experiment in Section 4.4.2 and is a common approach to retrieval tasks [17, 27].

4 Experiments

We follow the same evaluation protocol commonly used in image retrieval tasks with the standard train/test split on four datasets: CARS-196 [12], CUB-200-2011 [25], Stanford Online Products (SOP) [18], and In-shop Clothes Retrieval [34]. We compare our method using Recall@K to measure retrieval quality. To compute Recall@K, we use cosine similarity to retrieve the top K images from the test set, excluding the query image itself.

We first investigate how our embeddings trained with normalized softmax loss compare against embeddings trained with existing metric learning losses using the same featurizer and embedding dimension. We then study in detail how the dimensionality of our embeddings (Section 4.3) affects its performance and the relative performance between float and binary embeddings. Ablation studies on different design choices of our approach on CUB-200-2011 [25] (Section 4.4) are provided as empirical justifications. Next, we investigate how our embeddings are affected by class subsampling in Section 4.5, addressing the key scalability concern of softmax loss where training complexity is linear with the number of classes. Finally in Section 4.6, we show that our method outperforms state-of-the-art methods on several retrieval datasets.

4.1 Implementation

All experiments were executed using PyTorch and a Tesla V100 graphic card. We compare our method with common architectures used in metric learning including GoogleNet [20], GoogleNet with Batch Normalization [8], and ResNet50 [4]. We initialize our networks with pre-trained ImageNet ILSVRC-2012 [3] weights. We add a randomly initialized fully connected layer to the pool5 features of each architecture to learn embeddings of varying dimensionality. To simplify the sensitivity to the initialization of the fully connected layer, we add a Layer Normalization [10] without additional parameters between the pool5 and fully connected layer (See Section 4.4.1 for the ablation study). We L2 normalize the embedding and class weights before Softmax and use a temperature of 0.05 consistently.

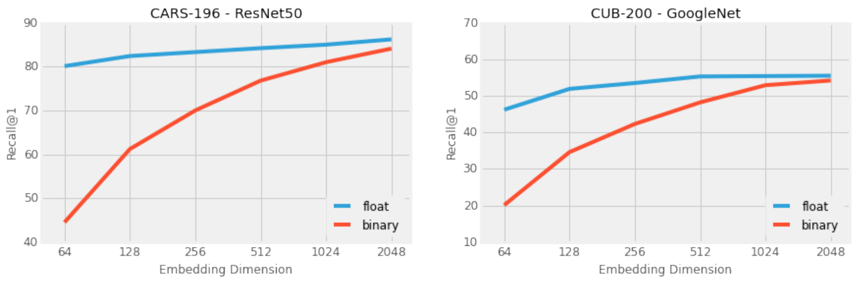

Figure 2: Recall@1 for CARS-196 (left) and CUB-200-2011 (right) across varying embedding dimensions. Softmax based embeddings improve performance when increasing dimensionality. The performance gap between float and binary embeddings converge when increasing dimensionality, showing that when given enough representational freedom, Softmax learns bit like features.

Unless otherwise stated, we first train for 1 epoch, updating only new parameters for better initialization. We then optimize all parameters for 30 epochs with a batch size of 75 with learning rate 0.01 with exception to CUB200 where we use learning rate of 0.001 due to overfitting. We construct our batch by sampling 25 examples per class for Cars196 and CUB200 and 5 examples per class for SOP and InShop (as only $\approx$ 5 images per class in dataset). We use SGD with momentum of 0.9, weight decay of 1e-4, and gamma of 0.1 to reduce learning rate at epoch 15. During training, we apply horizontal mirroring and random crops from 256x256 images; during testing we center crop from the 256x256 image. Following [9] [14], we crop to 227x227 for GoogleNet, otherwise 224x224. Complete implementation details can be found in our source code repository.[1]

4.2 Loss Function Comparisons

We compare our normalized softmax loss against existing metric learning losses. To focus on contributions from the loss functions, we leave comparisons against methods that ensemble models [29, 30], modify the feature extractor architecture [26], or propose complex activation paths between the featurizer and final embedding [15] for Section 4.6.

We present Recall@K and NMI results on three standard retrieval datasets in Table 1, Table 2, and Table 3, comparing against reported performance of methods trained with model architectures of GoogleNet, GoogleNet with Batch Normalization (BNInception), and ResNet50 respectively. For GoogleNet with Stanford Online Products only, we saw around a 1% Recall@1 improvement by training all parameters from start with 10x learning rate on new parameters when compared with models trained with our standard finetuning procedure.

As shown, our approach compares very strongly against existing baselines. When fixing dimensionality to 512, we see that the performance improvements of our softmax embeddings across architectures mirror classification performance on ImageNet ILSVRC-2012. We hope our results help disentangle performance improvements of existing metric learning methods due to advancements in methodology versus changes of base feature models.

	SOP				CARS-196					CUB-200				
Recall@K	1	10	100	NMI	1	2	4	8	NMI	1	2	4	8	NMI
Contras.^{128} [18]	42.0	58.2	73.8	-	21.7	32.3	46.1	58.9	-	26.4	37.7	49.8	62.3	-
Lift. Struc^{128} [18]	-	-	-	-	49.0	60.3	72.1	81.5	55.0	47.2	58.9	70.2	80.2	55.6
Lift. Struc^{512} [18]	62.1	79.8	91.3	-	-	-	-	-	-	-	-	-	-	-
Hist Loss^{512} [21]	63.9	81.7	92.2	-	-	-	-	-	-	50.3	61.9	72.6	82.4	-
Bin. Dev^{512} [21]	65.5	82.3	92.3	-	-	-	-	-	-	52.8	64.4	74.7	83.9	-
Npairs^{512} [17]	67.7	83.8	93.0	88.1	-	-	-	-	-	-	-	-	-	-
Npairs^{64} [17]	-	-	-	-	71.1	79.7	86.5	91.6	64.0	51.0	63.3	74.3	83.2	60.4
Angular^{512} [9]	**70.9**	**85.0**	**93.5**	**88.6**	71.4	81.4	87.5	92.1	63.2	54.7	66.3	76.0	83.9	61.1
NormSoftmax^{512}	69.0	84.5	93.1	88.2	**75.2**	**84.7**	**90.4**	**94.2**	**64.5**	**55.3**	**67.0**	**77.6**	**85.4**	**62.8**

Table 1: Recall@K and NMI across standard retrieval tasks. All methods are trained using GoogleNet for a fair comparison.

	SOP				CARS-196					CUB-200				
Recall@K	1	10	100	NMI	1	2	4	8	NMI	1	2	4	8	NMI
Clustering^{64} [19]	67.0	83.7	93.2	89.5	58.1	70.6	80.3	87.8	59.0	48.2	61.4	71.8	81.9	59.2
Proxy NCA^{64}[14]	73.7	-	-	**90.6**	73.2	82.4	86.4	88.7	64.9	49.2	61.9	67.9	72.4	59.5
HTL^{512} [24]	**74.8**	**88.3**	94.8	-	81.4	88.0	92.7	95.7	-	57.1	68.8	78.7	86.5	-
NormSoftmax^{512}	73.8	88.1	**95.0**	89.8	**81.7**	**88.9**	**93.4**	**96.0**	**70.5**	**59.6**	**72.0**	**81.2**	**88.4**	**66.2**

Table 2: Recall@K and NMI across standard retrieval tasks. All methods are trained using BNInception for a fair comparison

	SOP				CARS-196					CUB-200				
Recall@K	1	10	100	NMI	1	2	4	8	NMI	1	2	4	8	NMI
Margin^{128} [27]	72.7	86.2	93.8	90.7	79.6	86.5	91.9	95.1	69.1	**63.6**	**74.4**	83.1	**90.0**	69.0
NormSoftmax^{128}	75.2	88.7	95.2	90.4	81.6	88.7	93.4	96.3	72.9	56.5	69.6	79.9	87.6	66.9
NormSoftmax^{512}	**78.2**	**90.6**	**96.2**	**91.0**	**84.2**	**90.4**	**94.4**	**96.9**	**74.0**	61.3	73.9	**83.5**	**90.0**	**69.7**

Table 3: Recall@K and NMI across standard retrieval tasks. All methods are trained using ResNet50 for a fair comparison

4.3 Embedding Dimensionality

We study the effects of dimensionality on our classification-based embeddings by varying only the embedding dimension while keeping all other optimization parameters fixed. We have consistently observed that dimensionality is directly related to retrieval performance. Two examples of this across different datasets (CARS-196 and CUB-200-2011) and model architectures (ResNet50 and GoogleNet) are shown in Figure 2. Interestingly, this contradicts to reported behaviors for previous non-parametric metric learning methods [9, 18, 19], showing that dimensionality does not significantly affect retrieval performance. This difference is seen clearly when comparing R@1 across dimensionality for CUB-200-2011 with GoogleNet in Figure 2 with the same dataset and model combination from [18].

Higher dimensional embeddings lead to an increase in retrieval performance. Lower dimensional embeddings however are preferred for scalability to reduce storage and distance computation costs especially in large scale applications such as visual search [11]. We observe however that as we increase dimensionality of our embeddings, the optimizer does not fully utilize the higher dimensional metric space. Instead, we see that feature dimensions start relying less on the magnitude of each feature dimension and instead rely on the sign value. In Figure 2, we see for both datasets that the Recall@1 performance of binary features (thresholding the feature value at zero) converges with the performance of the float embeddings. This is a consistent result we see across datasets and model architectures. We show that training high dimensional embeddings and binarizing leads to the best trade-off of

[1]Code: https://github.com/azgo14/classification_metric_learning.git

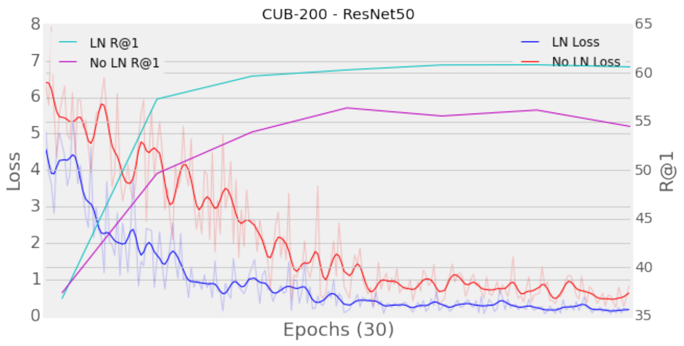

Figure 3: Loss and R@1 trends for training CUB-200 ResNet50 with and without Layer Normalization. Layer Normalization helps initialize learning, leading to better training convergence and R@1 performance.

S	-	1	3	12	25	37	75
C	-	75	25	6	3	2	1
R@1	59.5	59.6	60.0	60.8	61.3	59.6	40.9

Table 4: ResNet50 Recall@1 on CUB-200-2011 dataset across varying samples per class for batch size of 75. (**S**) Samples per class in batch. (**C**) Distinct classes in batch. First column shows no class balancing in batch

performance and scalability as described in Section 4.6.

4.4 Ablation Studies

In this set of experiments we report the effect on Recall@1 with different design choices in our approach on CUB-200-2011. We train the ResNet50 with embedding dimension of 512 variant as in Table 3. We fix all hyperparameters besides the one of interest.

4.4.1 Layer Normalization

We utilize Layer Normalization [10] without parameters to standardize activation values after pooling to help the initialization of our training. With 100 classes in CUB-200-2011, we expect that a random classifier would have a loss value of roughly $-\ln(\frac{1}{100}) = 4.6$. As shown in Table 3, this occurs when training with Layer Normalization, but not without. We have found incorporating Layer Normalization in our training allows us to be robust against poor weight initialization of new parameters across model architectures, alleviating the need for careful weight initialization.

4.4.2 Class Balanced Sampling

As seen in Table 4, class balanced sampling is beneficial over random sequential iteration over the dataset. Generally we've observed that performance improves when we have more samples per class until too few distinct classes exist in the minibatch where the bias of separating few distinct classes may introduce too much noise to the optimization, resulting

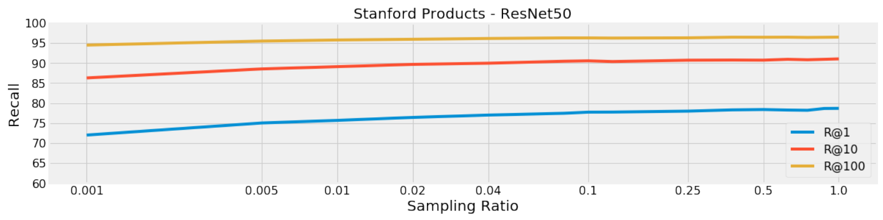

Figure 4: Recall@K for SOP with ResNet50 across class sampling ratios. We see that with sampling only 10% of classes per iteration (∼1K classes), we converge to a R@1 that is less than 1% absolute drop in performance from using all classes.

in lower performance. For SOP and InShop, we simply sample roughly how many images exist per class on average (≈ 5).

4.5 Subsampling for Classification Scalability

To show that classification-based training is scalable, we apply the subsampling methodology to the Stanford Online Products dataset, which has the largest number of classes among datasets we tested. By subsampling, for each training minibatch the network only need to classify among a subset of all classes (randomly sampled subset which includes all classes within the training minibatch). We present our findings in Figure 4, showing that with only 10% of the classes available during the forward pass of training, we can reach a R@1 performance comparable to using all classes (1% drop in performance). When using 1% of classes, we reach a R@1 of 75.7. When using 0.1% of classes, we reach a R@1 of 72.0. As we can see, subsampling classes during training is an effective method of scaling softmax embedding training with little drop in performance.

4.6 Comparison against State of the Art

Finally we compare our best performing softmax embedding model against state-of-the-art metric learning approaches on Stanford Online Products, In-Shop, CARS-196, and CUB-200-2011. We reproduced the state-of-the-art method in face verification literature, Large Maring Cosine Loss (LMCL) [23], and trained two variants of the networks: one with their recommended network architecture of 512 dimensional embeddings, and one with our modifications of 2048 dimensional embeddings.

As shown in Table 5 and Table 6, our 2048 dimensional ResNet50 embedding outperforms previous approaches. Considering the higher dimensionality of our embeddings, we also show that our 2048 **binary** embedding, sharing the same memory footprint of a 64 dimensional float embedding, similarly outperforms state-of-the-art baselines. These binary features were obtained by thresholding the float embedding features at zero as in Figure 2. Considering the scalability and performance of our binary softmax features along the simplicity of our approach, we believe softmax embeddings should be a strong baseline for future metric learning work.

	Net.	SOP			In-Shop			
Recall@K		1	10	100	1	10	20	30
Contrastive^{128} [18]	G	42.0	58.2	73.8	-	-	-	-
Lifted Struct^{512} [18]	G	62.1	79.8	91.3	-	-	-	-
Clustering^{64} [19]	B	67.0	83.7	-	-	-	-	-
Npairs^{512} [17]	G	67.7	83.8	93.0	-	-	-	-
HDC^{384} [30]	G	69.5	84.4	92.8	62.1	84.9	89.0	91.2
Angular Loss^{512} [9]	G	70.9	85.0	93.5	-	-	-	-
Margin^{128} [27]	R50	72.7	86.2	93.8	-	-	-	-
Proxy NCA^{64}[14]	B	73.7	-	-	-	-	-	-
A-BIER^{512} [15]	G	74.2	86.9	94.0	83.1	95.1	96.9	97.5
HTL^{512} [24]	B	74.8	88.3	94.8	-	-	-	-
HTL^{128} [24]	B	-	-	-	-	80.9	94.3	95.8
ABE-8^{512} [26]	G†	76.3	88.4	94.8	87.3	96.7	97.9	98.2
DREML^{9216} [29]	R18	-	-	-	78.4	93.7	95.8	96.7
LMCL^{512} [23]	R50	60.9	76.5	87.0	73.3	90.5	93.1	94.3
LMCL^{*2048} [23]	R50	77.4	89.7	95.3	**89.8**	**97.8**	98.6	98.8
NormSoftmax^{1024}	B	74.7	88.3	95.2	86.6	97.0	98.0	98.5
NormSoftmax^{128}	R50	75.2	88.7	95.2	86.6	96.8	97.8	98.3
NormSoftmax^{512}	R50	78.2	90.6	96.2	88.6	97.5	98.4	98.8
NormSoftmax^{2048}	R50	**79.5**	**91.5**	**96.7**	89.4	**97.8**	**98.7**	**99.0**
NormSoftmax$^{2048\text{bits}}$	R50	78.2	90.9	96.4	88.8	**97.8**	98.5	98.8

Table 5: Recall@K on Stanford Online Products (SOP) and In-Shop. R - ResNet, G - GoogleNet, B - BNInception, † refers to refers to additional attention parameters, LMCL* is our method trained with the Loss

	Net.	CARS-196				CUB-200			
Recall@K		1	2	4	8	1	2	4	8
Contrastive^{128} [18]	G	21.7	32.3	46.1	58.9	26.4	37.7	49.8	62.3
Lifted Struct^{128} [18]	G	49.0	60.3	72.1	81.5	47.2	58.9	70.2	80.2
Clustering^{64} [19]	B	58.1	70.6	80.3	87.8	48.2	61.4	71.8	81.9
Npairs^{64} [17]	G	71.1	79.7	86.5	91.6	51.0	63.3	74.3	83.2
Angular Loss^{512} [9]	G	71.4	81.4	87.5	92.1	54.7	66.3	76.0	83.9
Proxy NCA^{64} [14]	B	73.2	82.4	86.4	88.7	49.2	61.9	67.9	72.4
HDC^{384} [30]	G	73.7	83.2	89.5	93.8	53.6	65.7	77.0	85.6
Margin^{128} [27]	R50	79.6	86.5	91.9	95.1	63.6	74.4	83.1	90.0
HTL^{512} [24]	B	81.4	88.0	92.7	95.7	57.1	68.8	78.7	86.5
A-BIER^{512} [15]	G	82.0	89.0	93.2	96.1	57.5	68.7	78.3	86.2
ABE-8^{512} [26]	G†	85.2	90.5	94.0	96.1	60.6	71.5	79.8	87.4
DREML^{576} [29]	R18	86.0	91.7	95.0	97.2	63.9	75.0	83.1	89.7
LMCL^{512} [23]	R50	73.9	81.7	87.4	91.5	58.7	70.3	79.9	86.9
LMCL^{*2048} [23]	R50	88.3	93.1	95.7	97.4	61.2	71.4	80.4	87.4
NormSoftMax^{1024}	B	87.9	93.2	96.2	98.1	62.2	73.9	82.7	89.4
NormSoftmax^{128}	R50	81.6	88.7	93.4	96.3	56.5	69.6	79.9	87.6
NormSoftmax^{512}	R50	84.2	90.4	94.4	96.9	61.3	73.9	83.5	90.0
NormSoftmax^{2048}	R50	**89.3**	**94.1**	**96.4**	**98.0**	**65.3**	**76.7**	**85.4**	**91.8**
NormSoftmax$^{2048\text{bits}}$	R50	88.7	93.7	**96.4**	**98.0**	63.3	75.2	84.3	91.0

Table 6: Recall@K on CARS-196 and CUB-200-2011. R - ResNet, G - GoogleNet, B - BNInception, † refers to additional attention parameters, LMCL* is our method trained with the Loss

5 Conclusion

In this paper, we show that classification-based metric learning approaches can achieve state-of-the-art not only in face verification but general image retrieval tasks. In the metric learning community, a diverse set of base networks for training embedding of different sizes are compared to one another. In our work, we conducted comparisons through extensive experimentation, and establish that normalized softmax loss is a strong baseline in a wide variety of settings. We investigate common critiques of classification training and high dimensionality for metric learning through subsampling, making our approach viable even for tasks with a very large number of classes and binarization, allowing us to achieve SOTA performance with the same memory footprint as 64 dimensional float embeddings across the suite of retrieval tasks. We believe these practical benefits are valuable for large scale deep metric learning and real world applications.

References

[1] Sean Bell and Kavita Bala. Learning visual similarity for product design with convolutional neural networks. *ACM Trans. on Graphics (SIGGRAPH)*, 34(4), 2015.

[2] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546. IEEE, 2005.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[5] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In Defense of the Triplet Loss for Person Re-Identification. Technical report. URL https://arxiv.org/pdf/1703.07737.pdf.

[6] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. *CoRR*, abs/1412.6622, 2014. URL http://arxiv.org/abs/1412.6622.

[7] Houdong Hu, Yan Wang, Linjun Yang, Pavel Komlev, Li Huang, Xi (Stephen) Chen, Jiapei Huang, Ye Wu, Meenaz Merchant, and Arun Sacheti. Web-scale responsive visual search at bing. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 359–367, 2018. doi: 10.1145/3219819.3219843. URL http://doi.acm.org/10.1145/3219819.3219843.

[8] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. URL http://arxiv.org/abs/1502.03167.

[9] Shilei Wen Xiao Liu Yuanqing Lin Jian Wang, Feng Zhou. Deep metric learning with angular loss. In *International Conference on Computer Vision*, 2017.

[10] Geoffrey E. Hinton Jimmy Lei Ba, Jamie Ryan Kiros. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[11] Y. Jing, D. Liu, D. Kislyuk, A. Zhai, J. Xu, and J. Donahue. Visual search at pinterest. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (SIGKDD).*

[12] 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.

[13] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. *CVPR'17.*

[14] Yair Movshovitz-Attias, Alexander Toshev, Thomas K. Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. *CoRR*, abs/1703.07464, 2017. URL `http://arxiv.org/abs/1703.07464`.

[15] M. Opitz, G. Waltner, H. Possegger, and H. Bischof. Deep Metric Learning with BIER: Boosting Independent Embeddings Robustly. *arXiv:cs/1801.04815*, 2018.

[16] 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.

[17] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1857–1865. Curran Associates, Inc., 2016.

[18] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[19] Hyun Oh Song, Stefanie Jegelka, Vivek Rathod, and Kevin Murphy. Deep metric learning via facility location. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[20] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014.

[21] Evgeniya Ustinova and Victor Lempitsky. Learning deep embeddings with histogram loss. In *Neural Information Processing Systems*, 2016.

[22] Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 2018.

[23] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Zhifeng Li, Dihong Gong, Jingchao Zhou, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. *CVPR '18.*

[24] Dengke Dong Weifeng Ge, Weilin Huang and Matthew R. Scott. Deep metric learning with hierarchical triplet loss. In *ECCV*, 2018.

[25] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

[26] Kunal Chawla Jungmin Lee Keunjoo Kwon Wonsik Kim, Bhavya Goyal. Attention-based ensemble for deep metric learning. In *ECCV*, 2018.

[27] Chao-Yuan Wu, R. Manmatha, Alexander J. Smola, and Philipp Krähenbühl. Sampling matters in deep embedding learning. *CoRR*, abs/1706.07567, 2017.

[28] Zhirong Wu, Alexei A. Efros, and Stella X. Yu. Improving generalization via scalable neighborhood component analysis. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VII*, pages 712–728, 2018.

[29] Hong Xuan, Richard Souvenir, and Robert Pless. Deep randomized ensembles for metric learning. In *ECCV*, 2018.

[30] Yuhui Yuan, Kuiyuan Yang, and Chao Zhang. Hard-aware deeply cascaded embedding. *arXiv preprint arXiv:1611.05720*, 2016.

[31] Jingbin Wang David Tsai Chuck Rosenberg Yushi Jing, Henry Rowley and Michele Covell. Google image swirl: a large-scale content-based image visualization system. In *Proceedings of the 21st International Conference on World Wide Web*, 2012.

[32] Andrew Zhai, Dmitry Kislyuk, Yushi Jing, Michael Feng, Eric Tzeng, Jeff Donahue, Yue Li Du, and Trevor Darrell. Visual discovery at pinterest. *arXiv preprint arXiv:1702.04680*, 2017.

[33] Yanhao Zhang, Pan Pan, Yun Zheng, Kang Zhao, Yingya Zhang, Xiaofeng Ren, and Rong Jin. Visual search at alibaba. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 993–1001, 2018. doi: 10.1145/3219819.3219820. URL http://doi.acm.org/10.1145/3219819.3219820.

[34] Shi Qiu Xiaogang Wang Ziwei Liu, Ping Luo and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.