

Deep Metric Learning Meets Deep Clustering: An Novel Unsupervised Approach for Feature Embedding ¹

Binh X. Nguyen¹
binh.xuan.nguyen@aioz.io

¹ AIOZ
Singapore

Binh D. Nguyen¹
binh.duc.nguyen@aioz.io

² University of Liverpool
United Kingdom

Gustavo Carneiro³
gustavo.carneiro@adelaide.edu.au

³ University of Adelaide
Australia

Erman Tjiputra¹
erman.tjiputra@aioz.io





Quang D. Tran¹
quang.tran@aioz.io

Thanh-Toan Do²
thanh-toan.do@liverpool.ac.uk

Abstract

Unsupervised Deep Distance Metric Learning (UDML) aims to learn sample similarities in the embedding space from an unlabeled dataset. Traditional UDML methods usually use the triplet loss or pairwise loss which requires the mining of positive and negative samples w.r.t. anchor data points. This is, however, challenging in an unsupervised setting as the label information is not available. In this paper, we propose a new UDML method that overcomes that challenge. In particular, we propose to use a deep clustering loss to learn centroids, i.e., pseudo labels, that represent semantic classes. During learning, these centroids are also used to reconstruct the input samples. It hence ensures the representativeness of centroids — each centroid represents visually similar samples. Therefore, the centroids give information about positive (visually similar) and negative (visually dissimilar) samples. Based on pseudo labels, we propose a novel unsupervised metric loss which enforces the positive concentration and negative separation of samples in the embedding space. Experimental results on benchmarking datasets show that the proposed approach outperforms other UDML methods.

1 Introduction

The objective of deep distance metric learning (DML) is to train a deep learning model that maps training samples into feature embeddings that are close together for samples that belong to the same category and far apart for samples from different categories    .

¹This work was partially supported by the Australian Research Council project FT190100525.

[26, 66, 67, 88, 40, 43, 52, 55]. Traditional DML approaches require supervised information, i.e., class labels, to supervise the training. The class labels are used either for training the pointwise softmax loss [0, 62] or mining positive and negative samples for training the pairwise or triplet losses [6, 13, 29, 80, 64, 67]. Although the supervised DML achieves impressive results on different tasks [4, 16, 64, 65, 46, 56], it requires large amount of annotated training samples to train the model. Unfortunately, such large datasets are not always available and they are costly to annotate for specific domains. That disadvantage also limits the transferability of supervised DML to new domain/applications which do not have labeled data. These reasons have motivated recent studies aiming at learning feature embeddings without annotated datasets [24, 27, 64] — unsupervised deep distance metric learning (UDML). Our study is in that same direction, i.e., learning embeddings from unlabeled data.

There are two main challenges for UDML. Firstly, how to define positive and negative samples for a given anchor data point, such that we can apply distance-based losses, e.g., pairwise loss or triplet loss, in the embedding space. Secondly, how to make the training efficient, given a large number of pairs or triplets of samples, in the order of $\mathcal{O}(N^2)$ or $\mathcal{O}(N^3)$, respectively, in which N is the number of training samples. In this paper, we propose a new method that utilizes deep clustering for deep metric learning to address the two challenges mentioned above.

Regarding the first challenge, given an anchor point, its transformed versions (obtained, for example, from data augmentation methods) can be used as positive samples [10, 64]. Alternatively, the Euclidean nearest neighbors from pre-trained features can be used for selecting positive samples [10]. The problem is more challenging for mining negative samples. In [18], the authors rely on manifold mining which uses the pre-trained features of samples to build a similarity graph and the mining is performed on the graph. Hence, this approach heavily depends on pre-trained features. In addition, building a full adjacency matrix for graph mining is costly, especially for large scale training datasets. Recently, in [64], the authors consider all other training samples within the same batch of the anchor point as negative samples (w.r.t. the anchor) — this means that no negative mining is performed at all. This approach likely contains false negatives, especially when the number of same class samples in a batch is large. Different from these approaches, we propose to learn pseudo labels for samples through a deep clustering loss. The pseudo labels are then used for negative mining.

Regarding the second challenge, due to the large number of pairs or triplets w.r.t. all training samples, traditional approaches reduce the space by only considering $\mathcal{O}(m^2)$ pairs or $\mathcal{O}(m^3)$ triplets within each training batch containing m samples. In our approach, the reliance on the pseudo labels allows us to leverage the idea of center loss [0, 64] which results in a training complexity of $\mathcal{O}(Km)$ for each batch, in which K is the number of the learned centroids (i.e., pseudo labels).

Our main contributions can be summarized as follows. (i) We propose an novel UDML approach with a novel loss function that consists of a deep clustering loss for learning pseudo class labels, i.e., clusters, where each cluster is represented by a centroid. (ii) To enhance the representativeness of centroids, the loss also contains a reconstruction loss term that penalizes high sample reconstruction errors from centroid representations, which will encourage visually similar samples to belong to the same cluster. (iii) Based on pseudo labels, we propose a novel center-based loss that encourages samples to be close to their augmented versions and far from the centroids from different clusters in the embedding space.

2 Related Work

Supervised deep metric learning. The supervised deep metric learning uses the label information to supervise training [6, 11, 13, 27, 28, 31, 32, 37, 39, 47, 48, 49, 50]. Generally, the common loss functions used in supervised metric learning can be divided into two types: classification-based losses and distance-based losses. The classification-based losses focus on maximizing the probability of correct labels predicted for every sample [6, 31, 47, 48]. Those methods have linear run-time training complexity $\mathcal{O}(NC)$ where N and $C < N$ represent the number of training samples and the number of classes, respectively. The distance-based losses [11, 28, 32, 39, 49, 50] focus on learning the similarity or dissimilarity among sample embeddings. Two common distance-based losses are the contrastive (pairwise) loss [11] and the triplet loss [7, 13]. The run-time training complexity of contrastive loss is $\mathcal{O}(N^2)$ while triplet loss is $\mathcal{O}(N^3)$, in which N is the number of training samples. Hence both losses suffer from slow convergence due to a large number of trivial pairs or triplets as the model improves.

In order to overcome the run-time complexity challenge, many studies focused on mining strategies aiming at finding informative pairs or triplets for the training [13, 27, 32, 37]. For example, in [13], the authors propose to use fast approximate nearest neighbor search for quickly identifying informative (hard) triplets for training, reducing the training complexity to $\mathcal{O}(N^2)$. In [27], the mining is replaced by the use of $P < N$ proxies, where a triplet is re-defined to be an anchor point, a similar and a dissimilar proxy – this reduces the training complexity to $\mathcal{O}(NP^2)$. Recently, in [7] the authors propose a center-based metric loss in which centers are fixed during training and the loss involves the distance calculations between feature embeddings and the centers, rather than between features, resulting in $\mathcal{O}(NC)$ complexity, in which C is the number of centers.

Unsupervised feature learning. The common approach to unsupervised feature learning is to learn intermediate features, i.e., latent features, that well represent the input samples. Representative works are Autoencoder-based methods [53, 44]. Another approach is to leverage the idea of clustering, such as in [17, 19], where features are learned with the objective that they can be clustered into balanced clusters and the clusters are well separated. In [3], the authors propose to jointly learn the parameters of a deep neural network and the cluster assignments of the resulting features. More precisely, the method iterates between the k-means clustering of the features produced by the network and the updating of network weights by predicting the cluster assignments as pseudo labels using a discriminative loss.

Another popular approach to unsupervised feature learning is to replace the labels annotated by humans by labels generated from the original data with data augmentation methods. For example, in [9] the authors train a deep network to discriminate between a set of surrogate classes. Each surrogate class is formed by applying a variety of transformations to a randomly sampled image patch. In [23], the authors train a deep network such that it minimizes the distance between the features that describe a reference image and their rotated versions. Similarly, in [10], the authors train a convnet to predict the rotation that is applied to the input image.

Unsupervised deep metric learning. The unsupervised deep metric learning can be considered as a special case of unsupervised feature learning. In this case, the networks are trained to produce deep embedding features from unlabeled data. Furthermore, the learned

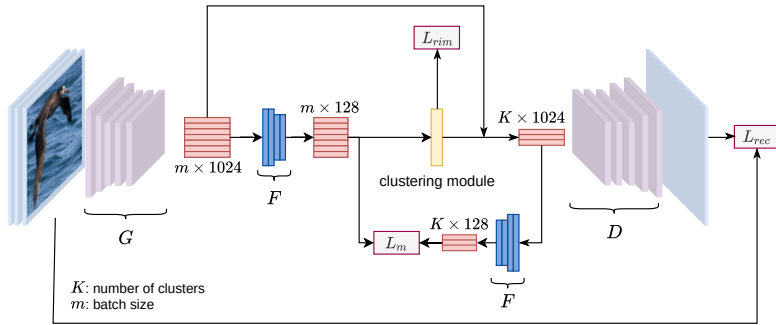


Figure 1: Illustration of the proposed framework which consists of an encoder (G), an embedding module (F), a decoder (D) and three losses, i.e., clustering loss L_{rim} , reconstruction loss L_{rec} and metric loss L_m . The details are presented in the text.

models are expected to generalize well to unseen testing class samples. Regarding this problem, there are only few works proposed to tackle the problem. In [13] the authors propose a mining strategy using Euclidean neighbors and manifold neighbors to pre-mine positive and negative pairs. The mining process involves building an Euclidean nearest neighbor graph from pre-computed features, a process that has $\mathcal{O}(N^2)$ complexity, in which N is the number of training samples. This means that the approach is not scalable and also heavily depends on the pre-computed features. To overcome the mining complexity, in [54], the authors propose a softmax-based embedding which enforces data augmentation invariance. Specifically, for each sample in a batch, its augmentations are considered as the positive samples and other samples are treated as negative ones. During training, the method needs to calculate the pairwise distance between samples in a batch, so its complexity for training a batch is quadratic w.r.t. batch size m .

3 Method

The proposed framework is presented in Figure 1. For every original image in a batch, we make an augmented version by using a random geometric transformation. Let the number of input images in a batch after augmentation be m . The input images are fed into the backbone network which is also considered as the encoder (G – purple) to get image representations (red) which have 1024 dimensions. The image representations are passed through the embedding module which consists of fully connected and $L2$ normalization layers (F – blue), which results in unit norm image embeddings with 128 dimensions. The clustering module (yellow) takes image embeddings as inputs, performs the clustering with a clustering loss (L_{rim}), and outputs the cluster assignments. Given the cluster assignments, centroid representations are computed from image representations, which are then passed through the decoder with a reconstruction loss (L_{rec}) to reconstruct images that belong to the corresponding clusters. The centroid representations are also passed through the embedding module (F) to get centroid embeddings. The centroid embeddings and image embeddings are used as inputs for the metric loss (L_m).

3.1 Discriminative Clustering

Following the idea of Regularized Information Maximization (RIM) for clustering [14, 19], we formulate the clustering of embedding features as a classification problem. Given a set of embedding features $X = \{x_i\}_{i=1}^m \in \mathbb{R}^{128 \times m}$ in a batch and the number of clusters $K \leq m$ (i.e., the number of clusters K is limited by the batch size m), we want to learn a probabilistic classifier, using a softmax activation, that maps each $x \in X$ into a probabilistic vector $y = \text{softmax}_\theta(x)$ that has K dimensions, in which θ is the classifier parameters¹. The cluster assignment for x then is estimated by $c^* = \arg \max_c y$.

Let $Y = \{y_i\}_{i=1}^m$ be the set of softmax outputs for X . Inspired by RIM [14, 19], we use the following objective function for the clustering

$$L_{rim} = \mathcal{R}(\theta) - \lambda [H(Y) - H(Y|X)], \quad (1)$$

where $H(\cdot)$ and $H(\cdot|X)$ are entropy and conditional entropy, respectively; $\mathcal{R}(\theta)$ regularizes the classifier parameters (in this work we use l_2 regularization); λ is a weighting factor to control the importance of two terms. Minimizing (1) is equivalent to maximizing $H(Y)$ and minimizing $H(Y|X)$. Increasing the marginal entropy $H(Y)$ encourages cluster balancing, while decreasing the conditional entropy $H(Y|X)$ encourages cluster separation [14]. Following [14] the entropy and the conditional entropy are calculated over the batch as follows

$$H(Y) = h\left(\frac{1}{m} \sum_{i=1}^m y_i\right), \quad (2)$$

$$H(Y|X) = \frac{1}{m} \sum_{i=1}^m h(y_i), \quad (3)$$

where $h(y_i) \equiv -\sum_{j=1}^K y_{ij} \log y_{ij}$ is the entropy function.

3.2 Reconstruction

In order to enhance the representativeness of centroids, we introduce a reconstruction loss that penalizes high reconstruction errors from centroids to corresponding samples. Specifically, the decoder takes a centroid representation of a cluster and minimizes the difference between input images that belong to the cluster and the reconstructed image from the centroid representation. By jointly training both the clustering loss and the reconstruction loss, we expect that centroids will well represent the samples belonging to the corresponding clusters.

Let X_j be a set of input images in the batch that belongs to the cluster j . The centroid representation for the cluster j is calculated as

$$r_j = \frac{1}{|X_j|} \sum_{I_i \in X_j} G(I_i), \quad (4)$$

where $G(\cdot)$ is the backbone network (i.e., the encoder), which extracts image representations. After obtaining the centroid representations, the reconstruction loss function is calculated as

¹In this context, the softmax classifier consists of an FC layer parameterized by θ that projects the feature embedding x to a vector with K dimensions, i.e., the logits. Then the softmax function is applied on the logits to output a probabilistic vector.

$$L_{rec} = \frac{1}{m} \sum_{j=1}^K \sum_{I_i \in X_j} \|I_i - D(r_j)\|^2, \quad (5)$$

where $D(\cdot)$ is the decoder which reconstructs samples in the batch using their corresponding centroid representations and m is the number of images in the batch.

3.3 Metric Loss

For every original image I_i in a batch, let \hat{I}_i be its augmented version. Let $f_i \in \mathbb{R}^{128}$ and $\hat{f}_i \in \mathbb{R}^{128}$ be the image embeddings of I_i and \hat{I}_i , respectively. The proposed metric loss aims to minimize the distance between f_i and \hat{f}_i while pushing f_i far away from negative clusters².

Given the cluster representation r_j for cluster j (see Section 3.2), the centroid embedding c_j is obtained by $c_j = F(r_j) \in \mathbb{R}^{128}$, where $F(\cdot)$ is the embedding module. It is worth noting that both the image embedding f and centroid embedding c are unit norm. Let the index of the cluster that sample i belongs to be q , $1 \leq q \leq K$. In order to minimize the distance between f_i and \hat{f}_i , and maximize the distance between f_i and centroids that I_i does not belong to, we maximize the following ‘‘softmax’’ like function

$$l(I_i, \hat{I}_i) = \frac{\exp(f_i^T \hat{f}_i / \tau)}{\sum_{k=1, k \neq q}^K \exp(f_i^T c_k / \tau)}, \quad (6)$$

where τ is the temperature parameter [14]. Using a higher value for τ produces a softer probability distribution.

Because feature embeddings and centroid embeddings are unit norm, maximizing (6) minimizes the Euclidean distance between f_i and \hat{f}_i and maximizes the Euclidean distances between f_i and the clusters that I_i does not belong to.

In addition, to push f_i far away from a cluster j that I_i does not belong to, we maximize the following quantity, $\forall j \neq q$

$$l(I_i, c_j) = 1 - \frac{\exp(f_i^T c_j / \tau)}{\sum_{k=1}^K \exp(f_i^T c_k / \tau)}. \quad (7)$$

By maximizing (7) $\forall j \neq q$, we maximize the Euclidean distance between f_i and c_j while minimizing the Euclidean distance between f_i and the centroid that I_i belongs to, i.e., c_q . Combining (6) and (7), we maximize the following quantity for each sample i

$$l(i) = l(I_i, \hat{I}_i) \prod_{j=1, j \neq q}^K l(I_i, c_j). \quad (8)$$

Applying the negative log-likelihood to (8) and summing over all training samples in the batch, we minimize the metric loss function which is defined as follows

$$L_m = - \sum_i \log(l(I_i, \hat{I}_i)) - \sum_i \sum_{j=1, j \neq q}^K \log(l(I_i, c_j)). \quad (9)$$

²Hereafter, I_i and \hat{I}_i are interchangeable when calculating losses.

3.4 Final Loss

The network in Figure 1 is trained in an end-to-end manner with the following multi-task loss

$$L = \alpha L_m + \beta L_{rim} + \gamma L_{rec}, \quad (10)$$

where L_m is the center-based softmax loss for deep metric learning (9), L_{rim} is the clustering loss (1), and L_{rec} is the reconstruction loss (5).

Complexity. In our metric loss L_m , we calculate the distance between samples and centroids, resulting in a $\mathcal{O}(Km)$ complexity. Both clustering loss L_{rim} and reconstruction loss L_{rec} have $\mathcal{O}(m)$ complexity. Hence the overall asymptotic complexity for training one batch of the proposed method is $\mathcal{O}(Km)$.

4 Experiments

4.1 Implementation Details

Following existing methods [18, 27, 28, 39, 54], the pre-trained GoogLeNet [42] on ImageNet is used as the backbone network (G). The embedding module (F) is added after the last average pooling of the GoogLeNet. A softmax layer is used as the clustering module to output a probabilistic vector for each image embedding. The decoder (D) is a stack of deconvolutional layers in which each layer is followed by a batch norm layer and ReLU activation. The values of α , β , and γ in (10) are empirically set to 0.9, 0.3, 0.01, respectively. We adopt the SGD optimizer with momentum 0.9. To make a fair comparison to SME [54], similar to that work, we also use the batchsize 64 (i.e., 128 after augmentation). The value of the temperature τ in (6), (7) is set to 0.1. For the data augmentation to create positive samples, the original images are randomly cropped at size 224×224 with random horizontal flipping which is similar to [18, 54]. In the testing phase, as a standard practice [18], a single center-cropped image is used as input to extract the image embedding.

4.2 Dataset and Evaluation Metric

We conduct our experiments on two public benchmark datasets that are commonly used to evaluate DML methods, where we follow the standard experimental protocol for both datasets [13, 39, 40, 41]. The **CUB200-2011** dataset [45] contains 200 species of birds with 11,788 images, where the first 100 species with 5,864 images are used for training and the remaining 100 species with 5,924 images are used for testing. The **Car196** dataset [20] contains 196 car classes with 16,185 images, where the first 98 classes with 8,054 images are used for training and the remaining 98 classes with 8,131 images are used for testing. We report the K nearest neighbor retrieval accuracy using the Recall@K metric. We also report the clustering quality using the normalized mutual information (NMI) score [29].

4.3 Ablation Study

In this section, we investigate the impact of each loss term in the proposed method. We also investigate the effect of the number of centroids in the clustering module which affects the pseudo labels. Here we consider the work SoftMax Embedding (SME) [54] as the baseline

	R@1	R@2	R@4	R@8
SME [54]	46.2	59.0	70.1	80.2
only L_{rim}	40.6	52.9	65.7	77.5
CBS	47.3	59.1	70.5	80.2
CBSwR	47.5	59.6	70.6	80.5

Table 1: The impact of each loss component on the performance on CUB200-2011 dataset and the comparison to the baseline [54].

	R@1	R@2	R@4	R@8
SME [54]	41.3	52.3	63.6	74.9
only L_{rim}	35.7	47.5	59.8	71.0
CBS	42.2	54.0	65.4	76.0
CBSwR	42.6	54.4	65.4	76.0

Table 2: The impact of each loss component on the performance on Car196 dataset and the comparison to the baseline [54].

Dataset	SME	only L_{rim}	CBS	CBSwR
CUB200-2011	696	620	660	882
Car196	772	700	727	1025

Table 3: The training time (seconds) of different methods on CUB200-2011 and Car196 datasets with 20 epochs. The models are trained on a NVIDIA GeForce GTX 1080-Ti GPU.

model. We denote our model that is trained by using only clustering loss (1) as **only L_{rim}** and our model that is trained with both clustering and the metric losses (1) and (9) as **Center-based Softmax (CBS)**. In this experiment, the number of clusters are fixed to 32 for both CUB200-2011 and Car196 datasets. We also investigate the impact of the reconstruction in enhancing the representativeness of centroids. This model is denoted as **Center-based Softmax with Reconstruction (CBSwR)**, which is our final model.

Tables 1 and 2 present the comparative results between methods. The results show that using only the clustering loss, the accuracy is significantly lower than the baseline. However, when using the centroids from the clustering for calculating the metric loss (i.e., CBS), it gives the performance boost over the baseline (i.e., SME). Furthermore, the reconstruction loss enhances the representativeness of centroids, as confirmed by the improvements of CBSwR over CBS on both datasets.

Table 3 presents the training time of different methods on the CUB200-2011 and Car196 datasets. Although the asymptotic complexity of CBSwR for training one batch is $\mathcal{O}(Km)$, it also consists of a decoder part which affects the real training. It is worth noting that the decoder is only involved during training. During testing, our method has similar computational complexity as SME.

Table 4 presents the impact of the number of clusters K in the clustering loss on the CUB200-2011 dataset with our proposed model CBSwR (recall that the number of clusters K is limited by the batch size m). During training, the number of samples per clusters vary depending on batches and the number of clusters. At $K = 32$ which is our final setting, the number of samples per cluster varies from 2 to 11, on the average. The retrieval performance is just slightly different for the different number of clusters. This confirms the robustness of the proposed method w.r.t. the number of clusters.

Number of clusters	R@1	R@2	R@4	R@8
16	46.7	59.0	70.0	79.7
32	47.5	59.6	70.6	80.5
48	47.1	58.9	70.6	80.1
64	47.4	58.9	69.6	80.2

Table 4: The impact of the number of clusters of the final model CBSwR on the performance on CUB200-2011 dataset.

	NMI	R@1	R@2	R@4	R@8
Supervised Learning					
SoftMax	57.2	48.3	60.2	71.2	80.3
N-pair [69]	57.2	45.4	58.4	69.5	79.5
Triplet+smart mining [13]	59.9	49.8	62.3	74.1	83.3
Triplet+proxy [27]	59.5	49.2	61.9	67.9	72.4
Histogram [43]	-	50.3	61.9	72.6	82.4
Unsupervised Learning					
Cyclic [22]	52.6	40.8	52.8	65.1	76.0
Exemplar [9]	45.0	38.2	50.3	62.8	75.0
NCE [63]	45.1	39.2	51.4	63.7	75.8
DeepCluster [8]	53.0	42.9	54.1	65.6	76.2
MOM [18]	55.0	45.3	57.8	68.6	78.4
SME [54]	55.4	46.2	59.0	70.1	80.2
CBSwR (Ours)	55.9	47.5	59.6	70.6	80.5

Table 5: Clustering and Recall performance on the CUB200-2011 dataset.

	NMI	R@1	R@2	R@4	R@8
Supervised Learning					
SoftMax	58.4	62.4	73.0	80.9	87.4
N-pair [69]	57.8	53.9	66.8	77.8	86.4
Triplet+smart mining [13]	59.5	64.7	76.2	84.2	90.2
Triplet+proxy [27]	64.9	73.2	82.4	86.4	88.7
Histogram [43]	-	54.3	66.7	77.2	85.2
Unsupervised Learning					
Exemplar [9]	35.4	36.5	48.1	60.0	71.0
NCE [63]	35.6	37.5	48.7	59.8	71.5
DeepCluster [8]	38.5	32.6	43.8	57.0	69.5
MOM [18]	38.6	35.5	48.2	60.6	72.4
SME [54]	35.8	41.3	52.3	63.6	74.9
CBSwR (Ours)	37.6	42.6	54.4	65.4	76.0

Table 6: Clustering and Recall performance on the Car196 dataset.

4.4 Comparison to the State of the Art

We compare our method to the state-of-the-art unsupervised deep metric learning and unsupervised feature learning methods that have reported results on the benchmarking datasets CUB200-2011 and CARS196. They are the graph-based method **Cyclic** [22], Exemplar CNN with surrogate classes from image transformations **Exemplar** [9], feature learning with noise-contrastive estimation **NCE** [63], alternative feature learning and k-means clus-

tering **DeepCluster** [3], mining on manifold **MOM** [18] and **SoftmaxEmbedding — SME** [54]. Among them, MOM [18] and SME [54] are the only methods that claim for unsupervised deep metric learning. We also compare our method to fully supervised deep metric learning methods, including the **softmax** loss, the **N-pair** [49] loss, the **histogram** loss [43], the **triplet with proxies** [27] loss, **triplet with smart mining** [13] loss which uses the fast nearest neighbor search for mining triplets.

Table 5 presents the comparative results on CUB200-2011 dataset. In terms of clustering quality (i.e., NMI metric), the proposed method and the state-of-the-art UDML methods MOM [18] and SME [54] achieve comparable accuracy. However, in terms of retrieval accuracy R@K, our method outperforms other approaches. Our proposed method is also competitive to most of the supervised DML methods.

Table 6 presents the comparative results on Car196 dataset. Compared to unsupervised methods, the proposed method outperforms other approaches in terms of retrieval accuracy at all ranks of K . Our method is comparable to other unsupervised methods in terms of clustering quality.

5 Conclusion

We propose a new method that utilizes deep clustering for deep metric learning to address the two challenges in UDML, i.e., positive/negative mining and efficient training. The method is based on a novel loss that consists of a learnable clustering function, a reconstruction function, and a center-based metric loss function. Our experiments on CUB200-2011 and Car196 datasets show state-of-the-art performance on the retrieval task, compared to other unsupervised learning methods.

References

- [1] Hossein Azizpour, Ali Sharif Razavian, Josephine Sullivan, Atsuto Maki, and Stefan Carlsson. From generic to specific deep representations for visual recognition. In *CVPR Workshops*, 2015.
- [2] John S Bridle, Anthony JR Heading, and David JC MacKay. Unsupervised classifiers, mutual information and ‘phantom targets’. In *NIPS*, 1992.
- [3] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, 2018.
- [4] Weihua Chen, Xiaotang Chen, Jianguo Zhang, and Kaiqi Huang. Beyond triplet loss: a deep quadruplet network for person re-identification. In *CVPR*, 2017.
- [5] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, 2005.
- [6] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *CVPR*, 2019.
- [7] Thanh-Toan Do, Toan Tran, Ian Reid, Vijay Kumar, Tuan Hoang, and Gustavo Carneiro. A theoretically sound upper bound on the triplet loss for improving the efficiency of deep distance metric learning. In *CVPR*, 2019.

-
- [8] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *NIPS*, 2014.
 - [9] Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *TPAMI*, pages 1734–1747, 2015.
 - [10] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018.
 - [11] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, 2006.
 - [12] Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In *CVPR*, 2015.
 - [13] Ben Harwood, B. G. Vijay Kumar, Gustavo Carneiro, Ian D. Reid, and Tom Drummond. Smart mining for deep metric learning. In *ICCV*, 2017.
 - [14] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning Workshop*, 2014.
 - [15] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, pages 84–92. Springer, 2015.
 - [16] Junlin Hu, Jiwen Lu, and Yap-Peng Tan. Discriminative deep metric learning for face verification in the wild. In *CVPR*, 2014.
 - [17] Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. Learning discrete representations via information maximizing self-augmented training. In *ICML*, 2017.
 - [18] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum. Mining on manifolds: Metric learning without labels. In *CVPR*, 2018.
 - [19] Andreas Krause, Pietro Perona, and Ryan G Gomes. Discriminative clustering by regularized information maximization. In *NIPS*, 2010.
 - [20] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *ICCV Workshops*, 2013.
 - [21] B. G. Vijay Kumar, Gustavo Carneiro, and Ian Reid. Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions. In *CVPR*, 2016.
 - [22] Dong Li, Wei-Chih Hung, Jia-Bin Huang, Shengjin Wang, Narendra Ahuja, and Ming-Hsuan Yang. Unsupervised visual representation learning by graph-based consistent constraints. In *ECCV*, 2016.
 - [23] Kevin Lin, Jiwen Lu, Chu-Song Chen, and Jie Zhou. Learning compact binary descriptors with unsupervised deep neural networks. In *CVPR*, 2016.

- [24] Chaochao Lu and Xiaoou Tang. Surpassing human-level face verification performance on lfw with gaussianface. In *AAAI*, 2015.
- [25] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
- [26] Jonathan Masci, Davide Migliore, Michael M Bronstein, and Jürgen Schmidhuber. Descriptor learning for omnidirectional image matching. In *Registration and Recognition in Images and Videos*, pages 49–62. Springer, 2014.
- [27] Yair Movshovitz-Attias, Alexander Toshev, Thomas K. Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. In *ICCV*, 2017.
- [28] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *CVPR*, 2016.
- [29] Qi Qian, Rong Jin, Shenghuo Zhu, and Yuanqing Lin. Fine-grained visual categorization via multi-stage metric learning. In *CVPR*, 2015.
- [30] Filip Radenovic, Giorgos Tolias, and Ondrej Chum. CNN image retrieval learns from bow: Unsupervised fine-tuning with hard examples. In *ECCV*, 2016.
- [31] Rajeev Ranjan, Carlos Castillo, and Ramalingam Chellappa. L2 constrained softmax loss for discriminative face verification, October 3 2019. US Patent App. 15/938,898.
- [32] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *CVPR Workshops*, 2014.
- [33] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *ICML*, 2011.
- [34] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015.
- [35] Hailin Shi, Yang Yang, Xiangyu Zhu, Shengcai Liao, Zhen Lei, Weishi Zheng, and Stan Z Li. Embedding deep metric for person re-identification: A study against large variations. In *ECCV*, 2016.
- [36] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, 2016.
- [37] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *ICCV*, 2015.
- [38] K. Simonyan, A. Vedaldi, and A. Zisserman. Learning local feature descriptors using convex optimisation. *TPAMI*, 2014.
- [39] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *NIPS*, 2016.
- [40] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *CVPR*, 2016.

- [41] Hyun Oh Song, Stefanie Jegelka, Vivek Rathod, and Kevin Murphy. Deep metric learning via facility location. In *CVPR*, 2017.
- [42] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.
- [43] Evgeniya Ustinova and Victor S. Lempitsky. Learning deep embeddings with histogram loss. In *NIPS*, 2016.
- [44] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, 2008.
- [45] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD birds-200-2011 dataset. 2011.
- [46] Faqiang Wang, Wangmeng Zuo, Liang Lin, David Zhang, and Lei Zhang. Joint learning of single-image and cross-image representations for person re-identification. In *CVPR*, 2016.
- [47] Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 25(7):926–930, 2018.
- [48] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*, 2018.
- [49] Xinshao Wang, Yang Hua, Elyor Kodirov, Guosheng Hu, Romain Garnier, and Neil M Robertson. Ranked list loss for deep metric learning. In *CVPR*, 2019.
- [50] Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R Scott. Multi-similarity loss with general pair weighting for deep metric learning. In *CVPR*, 2019.
- [51] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, 2016.
- [52] Paul Wohlhart and Vincent Lepetit. Learning descriptors for object recognition and 3d pose estimation. In *CVPR*, 2015.
- [53] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, 2018.
- [54] Mang Ye, Xu Zhang, Pong C Yuen, and Shih-Fu Chang. Unsupervised embedding learning via invariant and spreading instance feature. In *CVPR*, 2019.
- [55] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *CVPR*, 2015.
- [56] Jiahuan Zhou, Pei Yu, Wei Tang, and Ying Wu. Efficient online local metric adaptation via negative samples for person re-identification. In *ICCV*, 2017.