Nearest Neighbor Matching for Deep Clustering

Zhiyuan Dang[1,2], Cheng Deng[1]*, Xu Yang[1], Kun Wei[1], Heng Huang[3,4]
[1]School of Electronic Engineering, Xidian University, Xi'an 710071, China; [2]JD Tech, Beijing 100176, China
[3]Department of Electrical and Computer Engineering, University of Pittsburgh, PA 15260, USA
[4]JD Finance America Corporation, Mountain View, CA 94043, USA
{zhiyuandang, chdeng.xd, xuyang.xd, weikunsk}@gmail.com, heng.huang@pitt.edu

Abstract

Deep clustering gradually becomes an important branch in unsupervised learning methods. However, current approaches hardly take into consideration the semantic sample relationships that existed in both local and global features. In addition, since the deep features are updated on-the-fly, relying on these sample relationships may construct more semantically confident sample pairs, leading to inferior performance. To tackle this issue, we propose a method called Nearest Neighbor Matching (NNM) to match samples with their nearest neighbors from both local (batch) and global (overall) levels. Specifically, for the local level, we match the nearest neighbors based on batch embedded features, as for the global one, we match neighbors from overall embedded features. To keep the clustering assignment consistent in both neighbors and classes, we frame consistent loss and class contrastive loss for both local and global levels. Experimental results on three benchmark datasets demonstrate the superiority of our new model against state-of-the-art methods. Particularly on the STL-10 dataset, our method can achieve supervised performance. As for the CIFAR-100 dataset, our NNM leads 3.7% against the latest comparison method. Our code will be available at https://github.com/ZhiyuanDang/NNM.

1. Introduction

Unsupervised learning approach becomes emerging recently since the expensive label acquisition. As an important branch of unsupervised learning, clustering methods have attracted more attention, which goal is that grouping the samples into clusters, such that similar samples into the same cluster while dissimilar ones into different clusters.

Traditional clustering methods, such as K-Means [28], Spectral Clustering [43], Nonnegative Matrix Factorization [2], have been widely applied in various tasks. However, these methods only focus on low-level information, leading to their suboptimal performance. With the aid of the impressive deep learning methods [26], deep clustering methods are proposed to non-linearly transform the original samples into a latent embedded space for successfully providing more effective features and promising performance. Although conducting cluster analysis with learnable deep learning representations shows the potential to benefit clustering on such unlabelled data, how to improve the semantic confidence of these clusters remains an open question.

There are some methods proposed to solve it. According to the data training strategy, current deep clustering approaches could be roughly divided into two categories: The first one (such as DEC [38], JULE [39], DAC [5], DeepCluster [3], DDC [4], DCCM [36]) usually iteratively evaluate the clustering assignment from the up-to-date model and supervise the network training processes by the estimated information; The second one (such as ADC [14], IIC [21], PICA [20], DCDC[10]) simultaneously learn both the

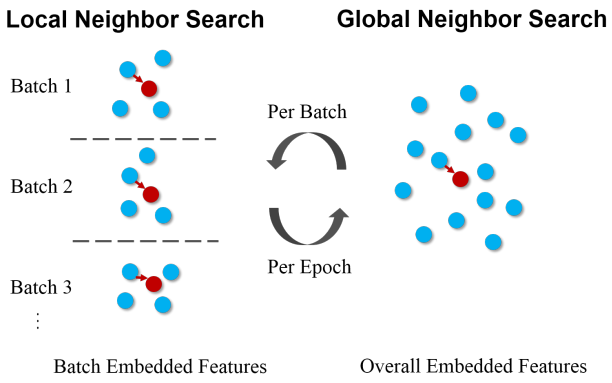


Figure 1. The illustration of our idea. We propose to match more semantically nearest neighbors from between local (batch) and global (overall) level. Benefit from the dynamic updated deep features with iteration and epoch increases, we can construct more and more semantically confident sample pairs from samples and its neighbors.

*Corresponding author.

feature representation and clustering assignment at the same time without explicit phases of clustering. Different from these previous studies, SCAN [34] is proposed to do semantic clustering with mining nearest neighbors. However, these approaches hardly take into consideration the semantic sample relationships existed in both local and global features. In addition, since the deep features are updated on-the-fly, relying on these sample relationships may construct more semantically confident sample pairs, leading to inferior performance.

As shown in Fig. 1, with the intuition that fully adopting the rich sample relationships existed in both local and global features, we plan to search the nearest neighbors from both these features. Note that we can easily construct semantic sample pairs from samples and its nearest neighbors. Therefore, in this paper, we propose a method named Nearest Neighbors Matching (NNM) for deep clustering. Specifically, for the local level, we match the nearest neighbors based on batch embedded features, as for the global, we match the neighbors from overall embedded features. To keep the clustering assignment consistent in both neighbors and classes, we adopt consistent loss and class contrastive loss for both local and global levels. Benefit from this novel NNM loss, our method can obtain more semantic clustering assignments. Significantly, our NNM is a plug-in module that can be applied in any works to obtain more semantic feature representations.

Our major contributions can be summarized as follows:

- We propose a novel deep clustering framework called NNM that is based on two-level nearest neighbors matching. Different from previous methods, NNM matches the nearest neighbors from local and global levels, and thus further improve clustering performance. In addition, our NNM is a plug-in module that can be adopted in any network to learn a more semantic feature representation.

- We provide the confusion matrices of three widely used benchmark datasets, after optimizing NNM loss. As shown in Fig. 4, desired confusion matrices should be block-diagonal. The most confident samples are shown together with the name of the matched ground-truth classes in Fig. 5. Besides that, we also conduct more ablation studies about numbers of clustering heads and nearest neighbors to figure out the useful techniques for current deep clustering.

- Extensive experimental results on three benchmark datasets demonstrate the superiority of our NNM against other state-of-the-art methods. Particularly on the STL-10 dataset, our method can achieve supervised performance. As for the CIFAR-100 dataset, our NNM leads 3.7% against the latest comparison method.

2. Related Work

In this section, we firstly review the latest deep clustering approaches and then give an introduction to current contrastive representation learning methods.

2.1. Deep Clustering

For alternate training based works, DEC [38] is a typical method that initializes clustering centroids by applying K-Means [27] on pre-trained image features and then fine-tunes the model to learn from the confident clustering assignments to sharpen the resulted prediction distribution. JULE [39] jointly optimizes the CNN in a recurrent manner, where merging operations of agglomerative clustering are conducted in the forward pass and representation learning is performed in the backward pass. DAC [5], DDC [4] and DCCM [36] alternately update the clustering assignment and inter-sample similarities during training. However, they are susceptible to the inconsistent estimations in the neighborhoods and thus suffer from severe error-propagation problem at training.

For current simultaneous training works, they more likes combing deep representation learning [1, 9, 18, 29, 41, 40] with conventional cluster analysis [11, 27, 15, 45, 44] or other pretext objectives. For the latest works, they are mostly based on mutual information maximization of the clustering assignment. For example, IIC [21] and IMSAT [19] are proposed to learn a clustering assignment by maximizing the mutual information between an image and its augmentations. Since the vague connection between the training supervision and clustering objective, PICA [20] deals with this limitation by introducing a partition uncertainty index to quantify the global confidence of the clustering assignment so as to select the most semantically plausible separation. DCDC [10] proposes to do contrastive learning over both sample and class views for more generalizable representation features. SCAN [34] is firstly proposed to further improve clustering semantics by a two-step procedure that firstly learns semantic features and then adopts the obtained features as a prior in a deep clustering network.

2.2. Contrastive Representation Learning

Instead of matching an input data to a fixed target, in contrastive loss [13] formulations the target can be various on-the-fly during training and can be defined in terms of the data representation computed by a network. Based on this novel loss, contrastive learning becomes a hot topic on recent unsupervised learning works [37, 18, 30, 17, 16, 6].

According to the numbers of sample pairs used to train, current contrastive learning methods can be divided into three categories. The first one is end-to-end mechanism (such as [13, 30, 18, 17, 6]) which is common and back-propagation. It uses samples of the current mini-batch to

construct sample pairs. But the sample pair number is coupled with the mini-batch size, limited by the GPU memory size which also is challenged by large mini-batch optimization [12]. The second one is memory-bank based mechanism [37]. A memory bank consists of all the sample features of the total dataset. Since the samples in mini-batch are randomly sampled from the memory bank without back-propagation, therefore, this method supports large batch size. However, the sample features in the memory bank are updated asynchronously with current sample features of the DNN, and thus are sometimes less consistent. The third one is the latest momentum encoder based mechanism [16]. [16] proposed to encode samples on-the-fly by a momentum-updated encoder, and maintain a queue of sample features.

Latest representation learning networks, such as MoCo [16] and SimCLR [6], are proposed with an instance discrimination manner. To this end, they pursue to learn a feature representation Φ_{pre} that is invariant to an augmentation image, and thus coincidentally adopt contrastive loss. The widely adopted contrastive loss, InfoNCE [30], has been proved as a lower bound estimation of mutual information, therefore, maximizing it will achieve the goal of representation learning works, i.e., mutual information maximization. That also provides the theoretical guarantee of the superior performance of current methods.

3. Methodology

We present our approach in the following sections. First, we pre-train an unsupervised representation learning model with the latest contrastive learning loss. After training the model, we search the nearest neighbors according to the similarity between features. Then, we develop the Nearest Neighbor Matching (NNM) method to obtain a more semantic clustering assignment.

3.1. Unsupervised Representation Learning Model

Since the previous end-to-end deep clustering methods are sensitive to the network initialization, we consider to pre-train an unsupervised learning model and cluster over such semantic features. Next, we provide the detailed objective formulation of the two contrastive learning works (i.e., MoCo [16] and SimCLR [6]). Note that at this step, we denote the last classification head of the backbone network as contrastive head.

Assume that we have a sample s and its augmented version s' and their features u and u' respectively. For SimCLR work, we have the following forms:

$$\mathcal{L}_{simclr} = -\log \frac{\exp(sim(u_i, u_i')/\tau)}{\sum_{j=1}^{2B} \mathbf{1}_{j\neq i} \exp(sim(u_i, u_j')/\tau)}, \quad (1)$$

where $sim()$ is similarity function like Cosine Similarity, B is the batch size, τ is the temperature parameter to tune and $\mathbf{1}_{k\neq i}$ is a indication function when $k \neq i$, its value is 1, otherwise 0. Obviously in the Eq. 1, this loss have one positive sample and $(2B-1)$ negative samples. As proved in [30], more negative samples can obtain tighter lower bound of mutual information which means the performance of SimCLR is related to batch size.

Contrast to SimCLR, MoCo maintains a dynamic update queue q to store the features of previous iterations.

$$\mathcal{L}_{moco} = -\log \frac{\exp(sim(u, u')/\tau)}{\sum_{j=1}^{M} \exp(sim(u, k_j)/\tau)}, \quad (2)$$

where M is the queue size. Obviously, this loss has one positive sample and M negative samples. Since the queue does not require gradient, the size M could be large, which is useful for large dataset feature learning.

Regarding the experimental datasets are smaller, SimCLR is enough to learn better semantic feature representation. For larger datasets like ImageNet, it better to utilize MoCo loss, particularly with a limited performance machine. After training the pre-train model, we first search top-K nearest neighbors according to the features before the contrastive head of the model.

3.2. Nearest Neighbor Matching

The mined nearest neighbors are important semantic supervised information that could be viewed as positive samples of the original samples. Then, we feed these positive pairs (i.e., original and one of nearest neighbors samples) into the clustering network Φ_{cls}. We load the pre-trained weights of representation model Φ_{pre} without the contrastive head. At this step, we denote the last classification head of the backbone network as clustering head, the batch size as B and the class number as C.

Surely, we can simply apply InfoNCE loss to make samples and their neighbors closer, and other neighbors farther. However, since there exist samples of same class in mini-batch, directly adopting InfoNCE loss may cause performance degeneration. Therefore, we decide only to maximize the similarity between samples and their neighbors as following consistent loss:

$$\mathcal{L}_{consi} = -\log < p, \mathcal{N}_{pre}(p) >, \quad (3)$$

where $p \in \mathbb{R}^{B\times C}$ and $\mathcal{N}_{pre}(p) \in \mathbb{R}^{B\times C}$ are clustering assignment of the sample u and its neighbors $\mathcal{N}_{pre}(u)$. Note that $\mathcal{N}_{pre}$ indicates the neighbors from pre-trained unsupervised representation models. The $< \cdot, \cdot >$ is dot product operator and used to measure the similarity. Since the sample assignment should be totally different in each iteration, with this insight, we develop the class contrastive loss:

$$\mathcal{L}_{class} = -\log \frac{\exp(sim(q_i, \mathcal{N}_{pre}(q)_i)/\tau)}{\sum_{j=1}^{C} \exp(sim(q_i, \mathcal{N}_{pre}(q)_j)/\tau)}, \quad (4)$$

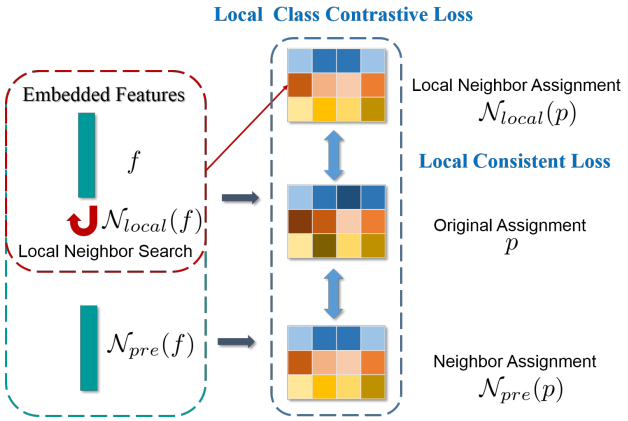


Figure 2. The illustration of the local NNM loss. Local consistent loss $\mathcal{L}_{local_{consi}}$ aims to keep both original, neighbor and local neighbor clustering assignments consistent. As for local class contrastive loss $\mathcal{L}_{local_{cla}}$, it wants to keep the sample view of these clustering assignments consistent.

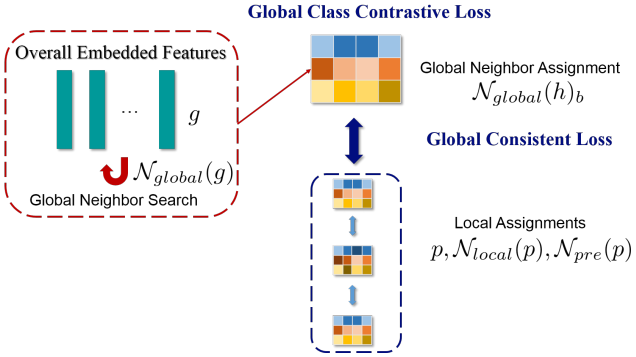


Figure 3. The illustration of the global NNM loss. Global consistent loss $\mathcal{L}_{global_{consi}}$ aims to keep both local and global neighbor clustering assignments consistent. As for global class contrastive loss $\mathcal{L}_{global_{cla}}$, it want to keep the sample view of these clustering assignments consistent.

where $q \in \mathbb{R}^{C\times B}$ and $\mathcal{N}_{pre}(q) \in \mathbb{R}^{C\times B}$ (q_i and $\mathcal{N}_{pre}(q)_i \in \mathbb{R}^B$) are the transpose of p and $\mathcal{N}_{pre}(p)$ respectively. To reduce the strength of constraint, τ in here can be set as 1. The dot product $\langle\cdot,\cdot\rangle$ in Eq. 3 provides more strong point-to-point-wise constraint on clustering assignment. As for $sim(\cdot,\cdot)$, we would like to make same prediction distribution in each class closer and other distributions far away, i.e., the class version of Eq.1 and Eq.2.

Besides that, we adopt a widely used entropy term [20, 34] to prevent the trivial solution that assign a majority of samples into a minority of clusters:

$$\mathcal{L}_{entropy} = -M(p)\log M(p). \tag{5}$$

where $M(p) \in \mathbb{R}^{1\times C}$ is the mean of p over batch dimension. A naive approach that mining nearest neighbors is based on the clustering assignment (i.e., p). Since there are several clustering heads in the models, their clustering assignment may be different, which may cause some error-propagation. A better way is to search nearest neighbors from embedded features (before the clustering head). In addition, with the network Φ_{cls} optimizing, the obtained features are updating on the fly. It is too regretful to ignore these useful features for mining more valuable neighbors. We pursue to mine these semantic neighbors from both local (batch) and global (overall) level in the following subsections. In this way, it is free to adopt multiple clustering heads strategy in our local and global NNM losses for robust prediction (See Section 4.5.2 for more details).

3.2.1 Local Nearest Neighbor Matching

Assume that $f \in \mathbb{R}^{B\times D}$ is the embedded features (before clustering head) of p, where D is the embedded feature dimension. We present the illustration of the local loss in Fig. 2.

For local-level, we select the nearest neighbors from f and denote as $\mathcal{N}_{local}(f)$ per batch. According to the batch index of $\mathcal{N}_{local}(f)$, we could obtain the corresponding neighbors of p as $\mathcal{N}_{local}(p)$. Then we could adopt same loss Eq. 3 to make their predictions consistent:

$$\mathcal{L}_{local} = -\log < p, \mathcal{N}_{local}(p) > . \tag{6}$$

Given the useful $\mathcal{N}_{pre}(p)$, we can concatenate them together along batch dimension, then rewrite the above loss as:

$$\mathcal{L}_{local_{consi}} = -\log < \begin{bmatrix} p \\ p \end{bmatrix}, \begin{bmatrix} \mathcal{N}_{local}(p) \\ \mathcal{N}_{pre}(p) \end{bmatrix} >, \tag{7}$$

where $\begin{bmatrix} p \\ p \end{bmatrix} \in \mathbb{R}^{2B\times C}$. Similarly, the class contrastive loss Eq. 4 can be rewritten as:

$$\mathcal{L}_{local_{cla}} = -\log \frac{\exp(sim(\begin{bmatrix} q \\ q \end{bmatrix}, \begin{bmatrix} \mathcal{N}_{local}(q) \\ \mathcal{N}_{pre}(q) \end{bmatrix}_i)/\tau)}{\sum_{j=1}^{C}\exp(sim(\begin{bmatrix} q \\ q \end{bmatrix}, \begin{bmatrix} \mathcal{N}_{local}(q) \\ \mathcal{N}_{pre}(q) \end{bmatrix}_j)/\tau)}, \tag{8}$$

where $\begin{bmatrix} q \\ q \end{bmatrix} \in \mathbb{R}^{C\times 2B}$.

3.2.2 Global Nearest Neighbor Matching

After a epoch, we obtain the latest overall dataset predictions $h \in \mathbb{R}^{N\times C}$ and features $g \in \mathbb{R}^{N\times D}$ where N is the training data size. We present the illustration of the global loss in Fig. 3. For global-level, we choose the nearest neighbors from g and denote its prediction as $\mathcal{N}_{global}(h)$ per epoch. Provided the batch index by p, we can obtain the predictions with batch size, i.e., $\mathcal{N}_{global}(h)_b \in \mathbb{R}^{B\times C}$. Then

Algorithm 1 Training procedure of NNM.

Input: Train data $\mathcal{X}$, train epochs T, iterations per epoch I, target class number C, hyper-parameter λ.

Pre-training the model with contrastive loss such as SimCLR and MoCo.
Obtaining the top-K neighbors of each sample according to the embedded features $\mathcal{N}_{pre}$.
Loading the pre-trained network weights without contrastive head.
for $t = 1, \ldots, T$ **do**
  for $i = 1, \ldots, I$ **do**
    Sampling a random mini-batch about images.
    Feeding the mini-batch images and one of their pre-trained neighbors into the model.
    Obtaining the embedded features f of each batch
    Computing the neighbors prediction $\mathcal{N}_{local}(p)$ and $\mathcal{N}_{pre}(p)$.
    if $\mathcal{N}_{global}(h)$ is available **then**
      Computing the overall loss (Eq.11).
    else
      Computing local sample consistent loss (Eq.7), class contrastive loss (Eq.8) and entropy loss (Eq.5).
    end if
  end for
  Obtaining the features g of each epoch.
  Computing the neighbors prediction $\mathcal{N}_{global}(h)$ and sent it to next epoch.
end for

Output: A deep clustering model Φ_{cls}.

the consistent loss Eq. 3 and class loss Eq. 4 are rewritten as:

$$\mathcal{L}_{global_{consi}} = -\log < \begin{bmatrix} p \\ \mathcal{N}_{local}(p) \\ \mathcal{N}_{pre}(p) \end{bmatrix}, \begin{bmatrix} \mathcal{N}_{global}(h)_b \\ \mathcal{N}_{global}(h)_b \\ \mathcal{N}_{global}(h)_b \end{bmatrix} >, \tag{9}$$

and

$$\mathcal{L}_{global_{cla}} = -\log \frac{\exp(sim(\begin{bmatrix} q \\ \mathcal{N}_{local}(q) \\ \mathcal{N}_{pre}(q) \end{bmatrix}, \begin{bmatrix} \mathcal{N}_{global}(h)_b \\ \mathcal{N}_{global}(h)_b \\ \mathcal{N}_{global}(h)_b \end{bmatrix}_i)/\tau)}{\sum_{j=1}^{C} \exp(sim(\begin{bmatrix} q \\ \mathcal{N}_{local}(q) \\ \mathcal{N}_{pre}(q) \end{bmatrix}, \begin{bmatrix} \mathcal{N}_{global}(h)_b \\ \mathcal{N}_{global}(h)_b \\ \mathcal{N}_{global}(h)_b \end{bmatrix}_j)/\tau)}. \tag{10}$$

respectively. By this way, we combine the valuable nearest neighbors from local and global level. Therefore, our goal is to minimize the following total loss function:

$$\begin{aligned} \mathcal{L}_{total} = \mathcal{L}_{global_{consi}} + \mathcal{L}_{global_{cla}} \\ + \mathcal{L}_{local_{consi}} + \mathcal{L}_{local_{cla}} + \lambda \mathcal{L}_{entropy}. \end{aligned} \tag{11}$$

where λ is the hyper-parameter. For simplicity, the total loss can also be rewritten as:

$$\mathcal{L}_{total} = \mathcal{L}_{global} + \mathcal{L}_{local} + \lambda \mathcal{L}_{entropy}. \tag{12}$$

where $\mathcal{L}_{global} = \mathcal{L}_{global_{consi}} + \mathcal{L}_{global_{cla}}$ and $\mathcal{L}_{local} = \mathcal{L}_{local_{consi}} + \mathcal{L}_{local_{cla}}$. The training procedure of NNM is summarized in Algorithm 1.

Datasets	Classes	Train Split	Test Split	Image Size
CIFAR-10	10	50,000	10,000	$32 \times 32 \times 3$
CIFAR-100	20	50,000	10,000	$32 \times 32 \times 3$
STL-10	10	5,000	8,000	$96 \times 96 \times 3$

Table 1. Statistics of three benchmark datasets.

4. Experiments

4.1. Datasets

In the experiments, we assess our NNM on three widely used object recognition datasets:

- **CIFAR-10 and CIFAR-100** [25]: A natural image dataset with 50,000/10,000 samples from 10(/100) classes for training and testing respectively.
- **STL-10** [7]: An ImageNet [33] sourced dataset containing 500/800 training/test images from each of 10 classes and additional 100,000 samples from several unknown categories.

We summarize the statistics of these datasets in Table. 1. For fair comparison, we adopt the same clustering setting as [34]: training on the train dataset and testing on the test dataset for CIFAR-10/100 and STL-10, and set the 20 superclasses as the ground truth of CIFAR-100.

4.2. Evaluation Metrics

There are three common clustering performance metrics in our experiments: Accuracy (**ACC**), Normalised Mutual Information (**NMI**) and Adjusted Rand Index (**ARI**). The predicted label is assigned by the dominating class label. All these metrics scale from 0 to 1 and higher values indicate better performance.

4.3. Experimental Setup

Our NNM is implemented by PyTorch 1.4.0 [31] and optimized by Adam [23] with 10^{-4} learning rate and 10^{-4} weight decay. We select a standard ResNet18 as the deep learning network backbone. For each sample, we mine the

Datasets	CIFAR-10			CIFAR-100			STL-10		
Metrics	NMI	ACC	ARI	NMI	ACC	ARI	NMI	ACC	ARI
K-Means [28]	0.087	0.229	0.049	0.084	0.130	0.028	0.125	0.192	0.061
SC [43]	0.103	0.247	0.085	0.090	0.136	0.022	0.098	0.159	0.048
AC [11]	0.105	0.228	0.065	0.098	0.138	0.034	0.239	0.332	0.140
NMF [2]	0.081	0.190	0.034	0.079	0.118	0.026	0.096	0.180	0.046
AE [1]	0.239	0.314	0.169	0.100	0.165	0.048	0.250	0.303	0.161
DAE [35]	0.251	0.297	0.163	0.111	0.151	0.046	0.224	0.302	0.152
DCGAN [32]	0.265	0.315	0.176	0.120	0.151	0.045	0.210	0.298	0.139
DeCNN [42]	0.240	0.282	0.174	0.092	0.133	0.038	0.227	0.299	0.162
VAE [24]	0.245	0.291	0.167	0.108	0.152	0.040	0.200	0.282	0.146
JULE [39]	0.192	0.272	0.138	0.103	0.137	0.033	0.182	0.277	0.164
DEC [38]	0.257	0.301	0.161	0.136	0.185	0.050	0.276	0.359	0.186
DAC [5]	0.396	0.522	0.306	0.185	0.238	0.088	0.366	0.470	0.257
ADC [14]	-	0.325	-	-	0.160	-	-	0.530	-
DDC [4]	0.424	0.524	0.329	-	-	-	0.371	0.489	0.267
DCCM [36]	0.496	0.623	0.408	0.285	0.327	0.173	0.376	0.482	0.262
IIC [21] (Best)	-	0.617	-	-	0.257	-	-	0.610	-
PICA [20] (Best)	0.591	0.696	0.512	0.310	0.337	0.171	0.611	0.713	0.531
DCDC [10]	0.585	0.699	0.506	0.310	0.349	0.179	0.621	0.734	0.547
Supervised	0.862	0.938	0.870	0.680	0.800	0.632	0.659	0.806	0.631
Pretext [6] + K-means	0.598	0.659	0.509	0.402	0.395	0.239	0.604	0.658	0.506
SCAN* [34] (Best)	0.715	0.816	0.665	0.449	0.440	0.283	0.673	0.792	0.618
NNM*	**0.748**	**0.843**	**0.709**	**0.484**	**0.477**	**0.316**	**0.694**	**0.808**	**0.650**

Table 2. The clustering performance on three challenging object image benchmarks after clustering ($*$) step. The best results of unsupervised method are indicated as **Bold**.

20 nearest neighbors based on current contrastive representation works (i.e., Step 2 in Algorithm 1). To speed up the training process, for local and global K nearest neighbor search, we set K is 1. For these smaller experimental datasets, we choose the SimCLR as the main loss function of the pre-trained model. The entropy term weight λ is set to 5. The images are strongly augmented by composing four randomly selected transformations from RandAugment [8] during the NNM clustering step. We search the nearest neighbors using the Faiss library [22]. All models perform the clustering step 100 epochs with 200 batch size and 3 clustering heads. For reproducibility, we initialize the network with fixed random seed as [36] and load the pre-trained models weights without clustering head.

4.4. Compared with the stat-of-the-art methods

We report the clustering results of SOTA methods including traditional methods (such as K-Means [28], SC [43], AC [11], NMF [2]), deep learning based approach (such as AE [1], DAE [35], DCGAN [32], DeCNN [42], VAE [24], JULE [39], DEC [38], DAC [5], ADC [14], DDC [4], DCCM [36], IIC [21], PICA [20], DCDC[10]) and latest pre-trained features based methods (such as SCAN [34]) at Table. 2. For comparison of methods performance, we directly cite the best results reported in related papers. We also provide the results of K-Means on the pretext features and fully-supervised manner results.

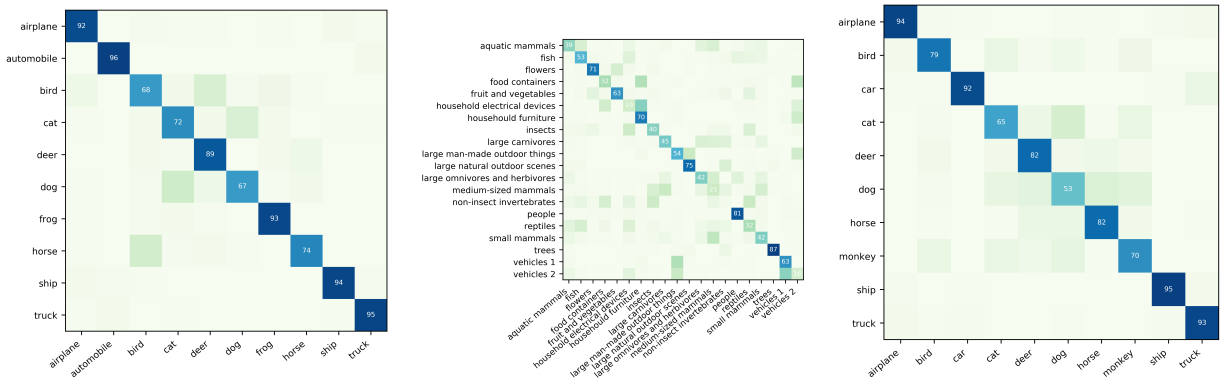
Benefit from the local and global nearest neighbors matching, our method outperforms other works on the three metrics. Particularly in CIFAR-100 dataset, NNM achieves 3.7% performance improvement. As for STL-10 dataset, the method can close to supervised performance. Due to the adopt of superclass of CIFAR-100, the best results of the proposed method and supervised results have large gap.

4.5. Ablation Study

In this section, we conduct several ablation studies to demonstrate the effect of different choices in NNM.

4.5.1 Effect of the proposed loss

Firstly, we assess the effect of the proposed loss respectively and provide the results in Table. 3. From the upper section of the Table, we state that the local loss $\mathcal{L}_{local}$ is more important to the overall loss than the global one. And the global loss $\mathcal{L}_{global}$ is more likely to refine the final results. For the results without local loss, we have some comments: actually, the global loss $\mathcal{L}_{global}$ works well or not depends on the learned feature of previous epochs, particularly in the first epoch, only entropy loss is available. Without lo-



(a) CIFAR-10 (b) CIFAR-100 (c) STL-10

Figure 4. Confusion matrices of three dataset.

Losses	NMI	ACC	ARI
NNM	**0.748**	**0.843**	**0.709**
$\mathbf{w/o}$ local loss $\mathcal{L}_{local}$	0.125	0.232	0.069
$\mathbf{w/o}$ global loss $\mathcal{L}_{global}$	0.736	0.835	0.692
$\mathbf{w/o}$ consistent loss $\mathcal{L}_{consis}$	0.718	0.826	0.673
$\mathbf{w/o}$ class contrastive loss $\mathcal{L}_{class}$	0.742	0.836	0.695
$\mathbf{w/o}$ entropy loss $\mathcal{L}_{entropy}$	0.326	0.204	0.152

Table 3. Effect of the proposed loss on CIFAR-10 dataset.

Datasets	Heads	NMI	ACC	ARI
CIFAR-10	1	0.748	0.842	0.707
	3	0.748	0.843	0.709
	5	**0.751**	**0.845**	**0.712**
CIFAR-100	1	**0.486**	0.471	**0.317**
	3	0.484	**0.477**	0.316
	5	0.473	0.467	0.313

Table 4. Effect of number of clustering heads on CIFAR-10 and CIFAR-100 datasets.

Global Nearest Neighbor	NMI	ACC	ARI
1	**0.694**	**0.808**	**0.650**
5	0.653	0.763	0.595
10	0.643	0.718	0.575

Table 5. Effect of number of global nearest neighbor on STL-10 dataset.

cal loss, the learned features will have less discrimination per iteration and epoch. Therefore, only global loss can not obtain performance improvement.

As for lower section of the Table, we can found that entropy loss $\mathcal{L}_{entropy}$ successfully prevents the network into the trivial solution that assigning most samples into a single cluster. Compared with class contrastive loss $\mathcal{L}_{class}$, consistent loss $\mathcal{L}_{consistent}$ play a vital role in the overall performance improvement.

4.5.2 Effect of number of clustering heads

Following [21], to obtain more robust predictions, we adopt the multiple clustering heads strategy in our method. We present the various results at Table. 4. With the number of heads increases, the results may have a certain degree improvement. Since we share the same global features on different clustering heads, the increasing heads may introduces more various predictions which makes results unstable (see results of CIFAR-100). Therefore, we set the number of heads as 3.

4.5.3 Effect of number of global nearest neighbor

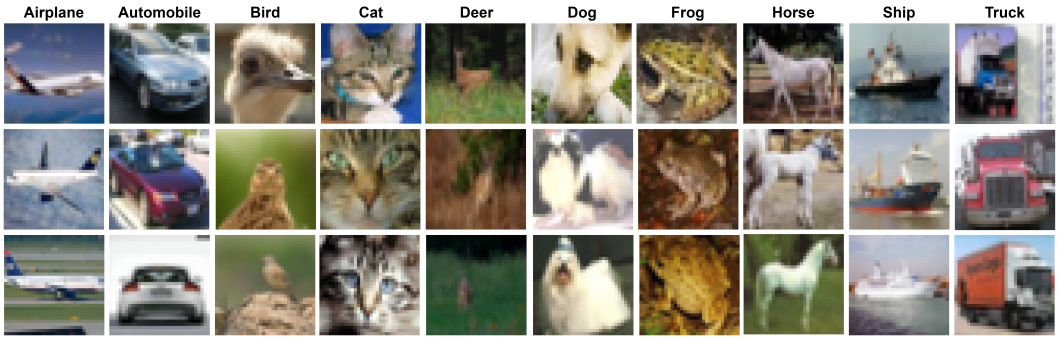
In this subsection, we also investigate the effect of number of global nearest neighbors and provide its results at Table. 5. Since the local features are updated faster than global ones, it is better to set the local neighbor as 1. Therefore, we only evaluate the performance with increasing global nearest neighbors. According to the presented results, the more global neighbors, the worse clustering performance. At the experiment of multiple neighbors, we randomly select a neighbor as the nearest one. Since the on-the-fly global features, more neighbors means that at one epoch the algorithm is more likely to select a noise neighbor, which will degenerate the performance.

4.6. Qualitative Study

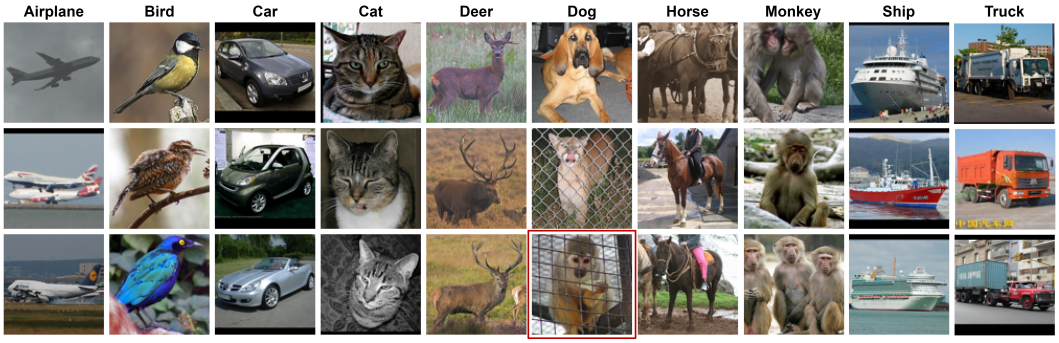
In this section, we conduct several qualitative studies to visually analyze the class confusion matrices and more confident samples.

4.6.1 Confusion Matrices

We visualize the confusion matrices of these three datasets at Fig. 4. These three confusion matrices have a clear block



(a) CIFAR-10



(b) STL-10

Figure 5. Top-3 most confident samples images on CIFAR-10 and STL-10.

diagonal structure which means our method successfully classify samples into semantic clusters. Most mis-clustering can be found that are hard to disentangle, even in visual cases, such as 'dog' and 'cat' in Fig. 4(a) and Fig. 4(c), 'household electrical devices' and 'household furniture' in Fig. 4(b). As the reason, on the one hand, some images are too small (32×32 in CIFAR-10/100) and blur to distinguish, and on the other hand, the network still can not disentangle some detailed fine-grained differences between some classes. We can obtain more intuitive feelings in next subsection.

4.6.2 Confident Images

We also visualize the different clusters after finishing training the model. Specifically, we provide the top-3 most confident samples in each cluster of CIFAR-10 and STL-10. Because of the poor performance of CIFAR-100, we does not report its confident samples. The results are shown together with the name of the matched ground-truth classes in Fig. 5. Importantly, we observe that the found samples align well with the classes of the dataset, except for the Dog class.

For the confident samples of Dog dataset (Fig. 5(b)), if we look with global view, the image more likes dog in a certain degree. That also indicates the network still can not disentangle some detailed fine-grained differences between several classes. Even so, the discriminative features of each object class are clearly presented in the images. Therefore, a direction of further improving clustering performance is to make the network figure out more fine-grained features on such datasets.

5. Conclusion

We have proposed a novel deep clustering framework based on mining more semantically nearest neighbors with local and global levels, named nearest neighbor matching (NNM). Different from previous methods, the NNM loss further improves clustering performance from local (batch) and global (overall) features. Benefiting from this novel loss, our network on three benchmark datasets outperforms state-of-the-art methods. Besides that, we also conduct more ablation studies about numbers of clustering heads and nearest neighbors to figure out the useful techniques for deep clustering.

Acknowledgment

Z.D., C.D., X.Y. and K.W. were supported in part by the National Natural Science Foundation of China under Grant 62071361, and the National Key R&D Program of China under Grant 2017YFE0104100, and CERNET Innovation Project under Grant NGII20190904.

References

[1] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *NeurIPS*, pages 153–160, 2007.

[2] Deng Cai, Xiaofei He, Xuanhui Wang, Hujun Bao, and Jiawei Han. Locality preserving nonnegative matrix factorization. In *IJCAI*, volume 9, pages 1010–1015, 2009.

[3] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, pages 132–149, 2018.

[4] Jianlong Chang, Yiwen Guo, Lingfeng Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Deep discriminative clustering analysis. *arXiv preprint arXiv:1905.01681*, 2019.

[5] Jianlong Chang, Lingfeng Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Deep adaptive image clustering. In *ICCV*, pages 5879–5887, 2017.

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[7] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *AISTATS*, pages 215–223, 2011.

[8] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020.

[9] Zhiyuan Dang, Cheng Deng, Xu Yang, and Heng Huang. Multi-scale fusion subspace clustering using similarity constraint. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6658–6667, 2020.

[10] Zhiyuan Dang, Cheng Deng, Xu Yang, and Heng Huang. Doubly contrastive deep clustering. *arXiv preprint arXiv:2103.05484*, 2021.

[11] K Chidananda Gowda and G Krishna. Agglomerative clustering using the concept of mutual nearest neighbourhood. *Pattern Recogn.*, 10(2):105–112, 1978.

[12] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[13] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, volume 2, pages 1735–1742. IEEE, 2006.

[14] Philip Haeusser, Johannes Plapp, Vladimir Golkov, Elie Aljalbout, and Daniel Cremers. Associative deep clustering: Training a classification network with no labels. In *GCPR*, pages 18–32. Springer, 2018.

[15] Zongbo Han, Changqing Zhang, Huazhu Fu, and Joey Tianyi Zhou. Trusted multi-view classification. In *International Conference on Learning Representations*, 2021.

[16] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9729–9738, 2020.

[17] Olivier J Hénaff, Aravind Srinivas, Jeffrey De Fauw, Ali Razavi, Carl Doersch, SM Eslami, and Aaron van den Oord. Data-efficient image recognition with contrastive predictive coding. *arXiv preprint arXiv:1905.09272*, 2019.

[18] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *ICLR*, 2018.

[19] Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. Learning discrete representations via information maximizing self-augmented training. In *International Conference on Machine Learning*, pages 1558–1567, 2017.

[20] Jiabo Huang, Shaogang Gong, and Xiatian Zhu. Deep semantic clustering by partition confidence maximisation. In *CVPR*, pages 8849–8858, 2020.

[21] Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *ICCV*, pages 9865–9874, 2019.

[22] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*, 2017.

[23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[24] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[25] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[26] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[27] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[28] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[29] Chuang Niu, Jun Zhang, Ge Wang, and Jimin Liang. Gatcluster: Self-supervised gaussian-attention network for image clustering. In *European Conference on Computer Vision*, pages 735–751. Springer, 2020.

[30] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[31] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[32] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[33] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[34] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Learning to classify images without labels. *arXiv preprint arXiv:2005.12320*, 2020.

[35] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(12), 2010.

[36] Jianlong Wu, Keyu Long, Fei Wang, Chen Qian, Cheng Li, Zhouchen Lin, and Hongbin Zha. Deep comprehensive correlation mining for image clustering. In *ICCV*, pages 8150–8159, 2019.

[37] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, pages 3733–3742, 2018.

[38] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *ICML*, pages 478–487, 2016.

[39] Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *CVPR*, pages 5147–5156, 2016.

[40] Xu Yang, Cheng Deng, Kun Wei, Junchi Yan, and Wei Liu. Adversarial learning for robust deep clustering. *Advances in Neural Information Processing Systems*, 33, 2020.

[41] Xu Yang, Cheng Deng, Feng Zheng, Junchi Yan, and Wei Liu. Deep spectral clustering using dual autoencoder network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4066–4075, 2019.

[42] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. Deconvolutional networks. In *CVPR*, pages 2528–2535. IEEE, 2010.

[43] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *NeurIPS*, pages 1601–1608, 2005.

[44] Changqing Zhang, Yajie Cui, Zongbo Han, Joey Tianyi Zhou, Huazhu Fu, and Qinghua Hu. Deep partial multi-view learning. *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[45] Changqing Zhang, Qinghua Hu, Huazhu Fu, Pengfei Zhu, and Xiaochun Cao. Latent multi-view subspace clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.