Jointly Learn the Base Clustering and Ensemble for Deep Image Clustering

Chen Liang
Anhui Provincial International Joint Research Center for Advanced Technology in Medical Imaging, School of Computer Science and Technology, Anhui University
Hefei, China
e21301210@stu.ahu.edu.cn

Zhiqian Dong
Anhui Provincial International Joint Research Center for Advanced Technology in Medical Imaging, School of Computer Science and Technology, Anhui University
Hefei, China
e22301269@stu.ahu.edu.cn

Sheng Yang
Anhui Provincial International Joint Research Center for Advanced Technology in Medical Imaging, School of Computer Science and Technology, Anhui University
Hefei, China
e22201048@stu.ahu.edu.cn

Peng Zhou
Anhui Provincial International Joint Research Center for Advanced Technology in Medical Imaging, School of Computer Science and Technology, Anhui University
Hefei, China
zhoupeng@ahu.edu.cn

***Abstract*—Deep image clustering attracts increasingly more attention in computer vision and multimedia communities. To tackle the stableness and robustness problems in clustering, clustering ensemble is applied to generate a better result by fusing multiple weak base clustering results. However, the existing deep clustering ensemble methods only focus on how to ensemble multiple *fixed* weak base results and ignore the influence of the consensus result on the base results. Alternatively, in this paper, we propose another question, i.e., how to use the ensemble to improve the base results? To this end, we present a novel joint deep clustering ensemble framework, which jointly generates the base results and does the ensemble, so that the deep clustering and the clustering ensemble can boost each other. In this framework, we design a base clustering generation module and an ensemble module and integrate them into a unified neural network architecture. The extensive experiments on benchmark datasets well demonstrate the effectiveness and superiority of the proposed method. The code is available at https://github.com/liangchen98/JDCE.**

***Index Terms*—Image clustering, deep clustering, clustering ensemble**

I. Introduction

Image clustering is a fundamental task in computer vision and multimedia. In recent years, since deep learning has achieved promising performance in many tasks, deep clustering for images has been widely studied [1]–[5]. These methods apply a deep neural network backbone, such as Convolutional Neural Network (CNN) and ResNet, to extract a semantic representation for each image, and then design a clustering layer for image clustering. For example, Xie et al. applied the AutoEncoder to extract representations and designed a KL-divergence minimization method for clustering [1]; Yang et al. used the CNN to learn the embedding and proposed a merging operation to obtain the final clustering results [3].

Although deep clustering often achieves promising performance, since clustering is an unsupervised learning task that cannot use the labels to guide the training, it often suffers from robustness and stableness problems [6], [7]. To address these issues, clustering ensemble has been proposed [8]–[15]. Clustering ensemble first generates multiple weak base clustering results and then fuses these base results to obtain a consensus one, which is often more robust and stable than the base ones. For example, Huang et al. fused the base results by a locally weighted method [13]; Zhou et al. learned a consensus result from multiple base kmeans with graph filter learning [16]. The above-mentioned methods are all shallow methods, which do not use neural networks to extract rich information from data. Most recently, few works focus on the deep clustering ensemble [7], [17]. For example, Huang et al. first applied a neural network to extract features and ensembled the base results obtained from the feature maps in each layer of the network to obtain a consensus result [7]; Metaxas et al. utilized the clustering ensemble to control the diversity of each cluster [17].

Notice that both [7] and [17] are two-step methods, i.e., they first use the neural networks to generate the base results, and then do the ensemble on the base results. The base results generation (i.e., the neural networks training) and the ensemble are two separate steps. However, since the base results are weak, which is one of the most important motivations of

Peng Zhou is the corresponding author.

clustering ensemble, *why not apply the ensemble to improve the base results and do the ensemble on the improved ones?* To this end, in this paper, we propose a novel Joint Deep Clustering Ensemble (JDCE) framework for image clustering, jointly generating base results and doing the ensemble.

The basic idea of this framework is that, on one hand, the deep neural network can learn more informative representations for generating base results and doing ensemble; on the other hand, the ensemble can in turn guide the neural network training to refine the representations and improve base results. Therefore, the base results generation and the ensemble can be boosted by each other. Notice that, this idea is significantly different from the above-mentioned clustering ensemble methods, including both the shallow ones and the deep ones. In their methods, the base results are *pre-given* or *fixed*, and they only focus on how to ensemble the fixed base results to learn a consensus result. Different from them, we propose an alternative idea of clustering ensemble, which is to utilize the ensemble to improve or refine the base results and try to obtain a better ensemble result from the better base results.

To fulfill this idea, we design a neural network with two modules: a base clustering generation module and an ensemble module. Since we focus on image clustering, in the base clustering generation module, we first use ResNet [18] to learn the representation of each image. Then, we feed the representations into a cluster head to generate multiple base clustering results. At last, in the ensemble module, we ensemble the multiple base results to a consensus one. We train the two modules in an iterative way, so that the ensemble can influence and guide the base clustering generation to improve the base results.

Our main contributions are summarized as follows:

- We propose a novel framework for deep clustering ensemble, i.e., applying the ensemble to improve the base results in turn.
- We design a new neural network to jointly generate multiple base results and do the ensemble.
- Extensive experiments show that the proposed method outperforms the deep image clustering methods and even the state-of-the-art deep clustering ensemble methods.

II. Joint Deep Clustering Ensemble

In this section, we introduce our JDCE method in more detail.

A. Overview

In image clustering, we aim to partition a set of N images into K clusters. To this end, we design a novel joint deep clustering ensemble framework JDCE, whose architecture is shown in Figure 1. It consists of two main modules, i.e., a base clustering generation module denoted as the purple dashed box in Figure 1 and an ensemble module denoted as the orange dashed box. Specifically, the base clustering generation module contains two parts: a representation learning backbone that can extract feature representations from images and a cluster head that generates multiple base clustering results from the learned representations. The ensemble module integrates base clusterings to obtain a consensus result. Moreover, the consensus result can in turn guide the representation learning and further improve the quality of the base clustering results.

B. Base Clustering Generation Module

Since we aim to handle images, we adopt ResNet [18] as the backbone network, which is denoted as the grey box in Figure 1. This backbone is pre-trained following the [19]. Then, we feed the representations into a cluster head denoted as the blue dashed box. The cluster head consists of two fully connected layers and a ReLu as the activation function.

For an image $\mathbf{X} \in \mathbb{R}^{H\times W\times C}$, where H, W, and C denote the height, width, and channel of the image, respectively, we extract the representation by using the backbone network $\mathcal{F}(\cdot)$, denoted as

$$\mathbf{f} = \mathcal{F}(X;\theta_{\mathcal{F}}) \tag{1}$$

where $\mathbf{f} \in \mathbb{R}^{128}$ is a 128-dimensional representation vector of image X, and $\theta_{\mathcal{F}}$ denotes the parameters in the network $\mathcal{F}(\cdot)$. Then, we feed $\mathbf{f}$ into the cluster head ($\mathcal{C}(\cdot)$) to obtain the clustering probability vector $\mathbf{y} = [p_1, \cdots, p_k] \in \mathbb{R}^K$, where p_i denotes the probability that image X belongs to the i-th cluster. This process can be denoted as:

$$\mathbf{y} = \mathcal{C}(\mathbf{f};\theta_{\mathcal{C}}) \tag{2}$$

where $\theta_{\mathcal{C}}$ denotes the parameters in $\mathcal{C}(\cdot)$. Then, we assign each image to the cluster with the highest probability.

Notice that, by using Eq.(2), we can only generate one base clustering result. Since we need the ensemble to improve the base results, we should generate multiple base results. To achieve this, we run the cluster head $\mathcal{C}$ multiple times with random dropout with different ratios to generate multiple base clustering results with diversity. To further increase the diversity among the base results, we design a base results selection method. In more detail, supposing we wish to ensemble M base results, we run $\mathcal{C}$ $2M$ times with different dropout ratios to generate $2M$ base results as a candidate set. Then, we select M base results from the candidate set according to their diversity. Here, we compute the Normalized Mutual Information (NMI) between each pair of the base results first. Notice that the lower the NMI is, the more different the two base results are. Then, given any base result, we compute the average NMI between it and all other base results as its diversity score. Finally, we select M results $\mathbf{Y}_1, \cdots, \mathbf{Y}_M \in \{0,1\}^{N\times K}$ with the lowest score as the base results for ensemble, where $(Y_k)_{ij} = 1$ means that in the k-th base result, the i-th image belongs to the j-th cluster, and $(Y_k)_{ij} = 0$ otherwise.

C. Ensemble Module

In the ensemble module, we need to ensemble the multiple base results $\mathbf{Y}_1, \cdots, \mathbf{Y}_M$ to a consensus result $\mathbf{Y}^*$. Notice that, we cannot directly ensemble them because of the unaligned problem in base clusterings [20]. For example, the

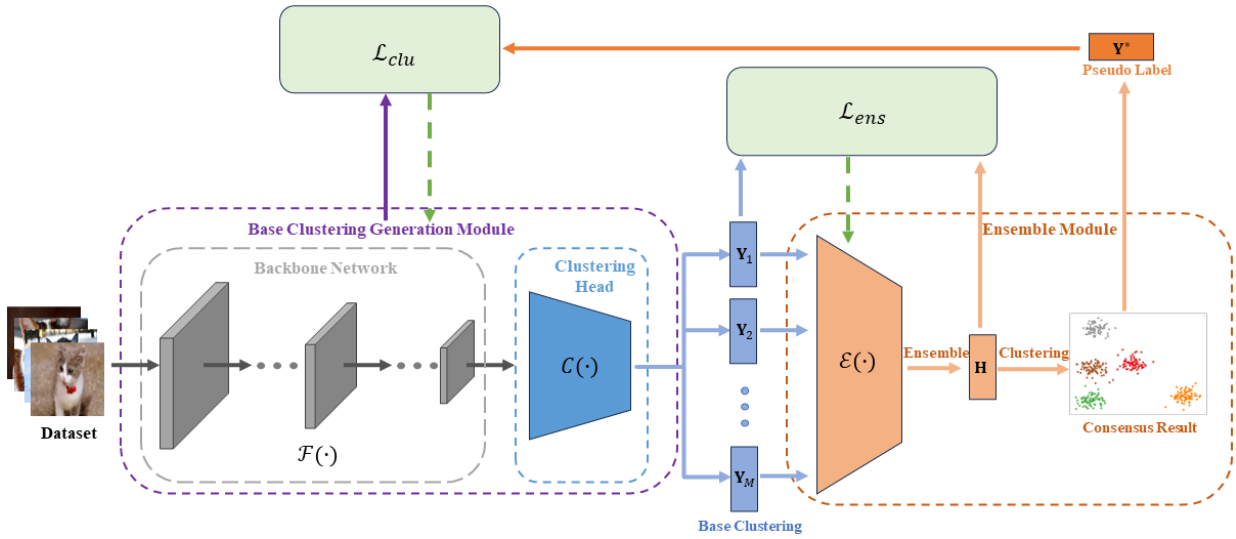


Fig. 1. The architecture of JDCE. It contains two modules: a base clustering generation module denoted as the purple dashed box, and an ensemble module denoted as the orange dashed box. Solid arrows mean the data flow and dashed arrows represent the gradient flow.

first cluster in $\mathbf{Y}_1$ is not necessarily to be the same as the first cluster in $\mathbf{Y}_2$. To tackle this problem, similar to the spectral rotation, we utilize a learnable orthogonal rotation matrix $\mathbf{R}_i \in \mathbb{R}^{K\times K}$ to align the clusters. Thus, $\mathbf{Y}_i\mathbf{R}_i$ is the i-th aligned base clustering which is ready to ensemble. Moreover, since the quality of each base result differs, we wish the better base results contribute more. To this end, we apply $0 \le \alpha_i \le 1$ as the weight of the i-th base result for the ensemble. Then, we obtain the clustering ensemble loss function $\mathcal{L}_{ens}$ as follows:

$$\begin{aligned} \min_{\mathbf{H},\mathbf{R}_i,\alpha} \quad & \mathcal{L}_{ens} = \sum_{i=1}^{M} \alpha_i^2 \|\mathbf{H} - \mathbf{Y}_i\mathbf{R}_i\|_F^2, \\ s.t. \quad & \mathbf{R}_i^T\mathbf{R}_i = \mathbf{I}, \quad \mathbf{H}^T\mathbf{H} = \mathbf{I}, \\ & 0 \le \alpha_i \le 1, \quad \sum_{i=1}^{M} \alpha_i = 1, \end{aligned} \tag{3}$$

where $\mathbf{I}$ denotes the identity matrix. $\mathbf{H} \in \mathbb{R}^{N\times K}$ is the consensus embedding of all images. We impose the orthogonal constraint on $\mathbf{H}$ because in the clustering task, we often wish each cluster to be far away from other clusters.

After obtaining the consensus embedding $\mathbf{H}$, we run k-means [21] on $\mathbf{H}$ to generate the consensus clustering result $\mathbf{Y}^* \in \{0,1\}^{N\times K}$, where $Y^*_{ij} = 1$ denotes the i-th image belongs to the j-th cluster and $Y^*_{ij} = 0$ otherwise.

When we obtain the consensus result $\mathbf{Y}^*$, we can apply it to improve the representation learning in the base clustering generation module. In more detail, we regard the $\mathbf{Y}^*$ as pseudo-labels and refine the backbone network $\mathcal{F}(\cdot)$ and cluster head $\mathcal{C}(\cdot)$ in a self-supervised way. Here, we design the self-supervised clustering loss $\mathcal{L}_{clu}$ with the cross-entropy loss ($\mathcal{L}_{CE}$) between the learned embedding $\mathcal{C}(\mathcal{F}(X,\theta_\mathcal{F}),\theta_\mathcal{C})$ and the consensus result $\mathbf{Y}^*$, which is shown as follows:

$$\min_{\theta_\mathcal{F},\theta_\mathcal{C}} \quad \mathcal{L}_{clu} = \mathcal{L}_{CE}(\mathcal{C}(\mathcal{F}(X,\theta_\mathcal{F}),\theta_\mathcal{C}),\mathbf{Y}^*) \tag{4}$$

D. Optimization

There are two groups of parameters in our method, i.e., $\theta_\mathcal{F}$ and $\theta_\mathcal{C}$ in the base clustering generation module and $\mathbf{H}$, $\mathbf{R}$, and α in the ensemble module. We optimize them iteratively. At first, we initialize $\mathbf{H} = \mathbf{0}$, $\mathbf{R}_i = \mathbf{I}$, $\alpha_i = \frac{1}{M}$, where $\mathbf{0}$ denotes the all-zero matrix. $\theta_\mathcal{F}$ and $\theta_\mathcal{C}$ are initialized by the pre-training. Then we first optimize the parameters in ensemble module i.e., Eq.(3).

Optimizing $\mathbf{H}$: Denoting $\mathbf{A}_i = \mathbf{Y}_i\mathbf{R}_i$, Eq.(3) can be rewritten as follows:

$$\begin{aligned} & \min_{\mathbf{H}^T\mathbf{H}=\mathbf{I}} \quad \sum_{i=1}^{M} \alpha_i^2 \|\mathbf{H} - \mathbf{A}_i\|_F^2 \\ = & \min_{\mathbf{H}^T\mathbf{H}=\mathbf{I}} \quad \sum_{i=1}^{M} \alpha_i^2 \left(tr(\mathbf{H}^T\mathbf{H}) - 2tr(\mathbf{H}^T\mathbf{A}_i) + tr(\mathbf{A}_i^T\mathbf{A}_i) \right) \\ = & \min_{\mathbf{H}^T\mathbf{H}=\mathbf{I}} \quad tr\left(\mathbf{H}^T \left(-2\sum_{i=1}^{M} \alpha_i^2 \mathbf{A}_i \right) \right) \end{aligned} \tag{5}$$

The second equation holds because $\mathbf{H}^T\mathbf{H} = \mathbf{I}$ and $\mathbf{A}_i^T\mathbf{A}_i$ is a constant. Now, we denote $\mathbf{A} = 2\sum_{i=1}^{M} \alpha_i^2 \mathbf{A}_i$, and denote the singular value decomposition (SVD) of $\mathbf{A}$ as $\mathbf{A} = \mathbf{U}_1\mathbf{\Sigma}_1\mathbf{V}_1^T$, where $\mathbf{U}_1$ and $\mathbf{V}_1$ are orthogonal matrices and $\mathbf{\Sigma}_1$ is a diagonal matrix. Then, according to Theorem 3 in [20], the closed-form solution of Eq.(5) is:

$$\mathbf{H} = \mathbf{U}_1\mathbf{V}_1^T \tag{6}$$

Optimizing $\mathbf{R}_i$: When fixing other variables, we can simplify Eq.(3) to:

$$\begin{aligned} \min_{\mathbf{R}_i} \quad & tr(-\alpha_i^2\mathbf{R}_i^T\mathbf{Y}_i^T\mathbf{H}), \\ s.t. \quad & \mathbf{R}_i^T\mathbf{R}_i = \mathbf{I}. \end{aligned} \tag{7}$$

Similar to the optimization of $\mathbf{H}$, we can get the SVD of $\alpha_i^2\mathbf{Y}_i^T\mathbf{H}$ as $\alpha_i^2\mathbf{Y}_i^T\mathbf{H} = \mathbf{U}_2\mathbf{\Sigma}_2\mathbf{V}_2^T$ and the closed-form solution of $\mathbf{R}_i$ is :

$$\mathbf{R}_i = \mathbf{U}_2\mathbf{V}_2^T \tag{8}$$

Optimizing α_i: By denoting $d_i = \|\mathbf{H} - \mathbf{Y}_i\mathbf{R}_i\|_F^2$, we obtain:

$$\begin{aligned} \min_{\alpha_i} \quad & \sum_{i=1}^{M} \alpha_i^2 d_i \\ s.t. \quad & 0 \le \alpha_i \le 1, \quad \sum_{i=1}^{M} \alpha_i = 1 \end{aligned} \tag{9}$$

According to the Cauchy-Buniakowsky-Schwarz Inequality, we can obtain the closed-form solution of α_i:

$$\alpha_i = \frac{d_i^{-1}}{\sum_{j=1}^{M} d_j^{-1}} \tag{10}$$

Optimizing $\theta_{\mathcal{F}}$ **and** $\theta_{\mathcal{C}}$: When optimizing $\theta_{\mathcal{F}}$ and $\theta_{\mathcal{C}}$, we fix the variables $\mathbf{H}$, $\mathbf{R}$, and α and optimize the clustering loss $\mathcal{L}_{clu}$. We first generate the consensus result $\mathbf{Y}^*$ by running k-means on $\mathbf{H}$ as the pseudo-labels. Then, we utilize Adam optimizer to optimize $\theta_{\mathcal{F}}$ and $\theta_{\mathcal{C}}$ by minimizing $\mathcal{L}_{clu}$.

The whole algorithm is summarized in Algorithm 1.

Algorithm 1 Joint Deep Clustering Ensemble

Input: Image set $\mathcal{X}$, number of iterations $maxIter$, training epochs E, size of base clusterings M.
Output: Final clustering results

```
Initialize H = 0, R_i = I, α_i = 1/M.
Feed all images to the pre-trained neural network.
for iteration = 1 to maxIter do
    Generate M base clusterings Y_1, ··· , Y_M.
    while not converge do
        Update H, R_i, and α_i by Eqs.(6), (8), and (10), respectively.
    end while
    Generate Y* from H.
    for training epoch = 1 to E do
        Update θ_F and θ_C with Adam optimizer.
    end for
end for
Generate Y by Eqs.(1) and (2).
Run k-means on Y to obtain the final clustering result.
```

III. Experiments

A. Datasets

We conduct experiments on 5 benchmark datasets, including CIFAR-10 [22], CIFAR-100 [22], STL-10 [23], ImageNet-10 [19], and ImageNet-Dogs [19]. In CIFAR-100, following [7], [17], [19], [24], we also use 20 super-classes rather than 100 classes. The details of these datasets are shown in Table I.

B. Implementation Details and Experiments Setup

We adopt ResNet34 as our backbone for representation learning. There are two fully connected layers in our cluster head, which can be mainly described as 128-128-K, where 128 is the dimension of features in hidden layers, and K is the number of clusters, which is predefined as the number of classes of each dataset. In the experiments, our model uses the same parameter setting on all datasets. We fix the batch size to 128. The Adam optimizer is applied with a learning rate of 0.0001 without weight decay. The number of base results M is 10. The number of iterations $maxIter$ is fixed to 5, and the number of epochs in each iteration E is fixed to 50. The experiments are conducted by Pytorch on an NVIDIA GeForce RTX 3090 24GB GPU.

TABLE I
The Detail of Datasets

Datasets	Size	Samples	Classes
CIFAR-10	32×32	60000	10
CIFAR-100	32×32	60000	20
STL-10	96×96	13000	10
ImageNet-10	96×96	13000	10
ImageNet-Dogs	96×96	19500	15

We compare our method with other representative traditional clustering methods, including k-means [21], SC [25], AC [26], and NMF [27]; and deep clustering methods, including AE [28], DAE [29], DeCNN [30], VAE [31], JULE [3], DEC [1], DAC [2], ADC [32], DCGAN [33], DDC [34], DCCM [35], IIC [36], PICA [24] and CC [19]. Moreover, we also compare against two state-of-the-art deep clustering ensemble methods, namely, DeepCluE [7] and DivClust [17].

To evaluate the clustering performance, we use three popular metrics: Normalized Mutual Information (NMI), Adjusted Rand Index (ARI), and clustering ACCuracy (ACC).

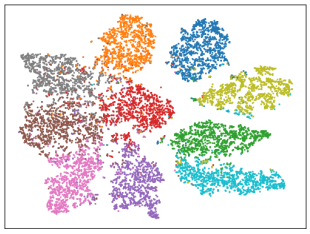
C. Experimental Results

Table II shows the ACC, NMI, and ARI results of all methods on all datasets. Especially, compared with both the conventional methods and deep clustering methods, our deep ensemble method can outperform them, which shows the effectiveness of the ensemble learning. Moreover, compared with the two state-of-the-art deep clustering ensemble methods DeepCluE and DivClust, ours also often achieves better performance, which demonstrates the superiority of the schema that jointly does the clustering and ensemble.

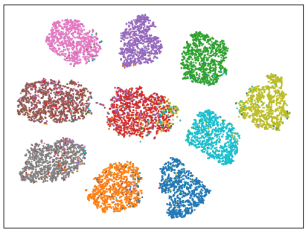
To further show the effects of the ensemble on the representation learning, we show the t-SNE visualization results on STL-10 and ImageNet-10 in Figure 2. Figure 2(a) and (c) show the clustering performances before the ensemble, and Figure 2(b) and (d) show the results after the ensemble. We can find that, after the ensemble, the learned representations display a clearer clustering structure, which demonstrates the motivation of our method, that is the ensemble is indeed helpful to learn a good representation to improve the base results.

TABLE II
THE CLUSTERING PERFORMANCE ON ALL DATASETS. RED TEXTS INDICATE THE BEST RESULTS, BLUE TEXTS INDICATE THE SECOND BEST RESULTS, AND GREEN TEXTS INDICATE THE THIRD BEST RESULTS.

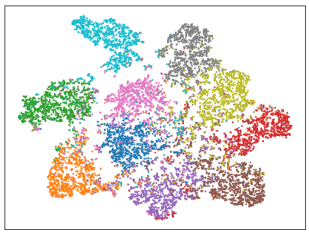
Datasets	**CIFAR-10**			**CIFAR-100**			**STL-10**			**ImageNet-10**			**ImageNet-Dogs**		
Metrics	***NMI***	***ACC***	***ARI***	***NMI***	***ACC***	***ARI***	***NMI***	***ACC***	***ARI***	***NMI***	***ACC***	***ARI***	***NMI***	***ACC***	***ARI***
K-means [21]	0.087	0.229	0.049	0.084	0.130	0.028	0.125	0.192	0.061	0.119	0.241	0.057	0.055	0.105	0.020
SC [25]	0.103	0.247	0.085	0.090	0.136	0.022	0.098	0.159	0.048	0.151	0.274	0.076	0.038	0.111	0.013
AC [26]	0.105	0.228	0.065	0.098	0.138	0.034	0.239	0.332	0.140	0.138	0.242	0.067	0.037	0.139	0.021
NMF [27]	0.081	0.190	0.034	0.079	0.118	0.026	0.096	0.180	0.046	0.132	0.230	0.065	0.044	0.118	0.016
AE [28]	0.239	0.314	0.169	0.100	0.165	0.048	0.250	0.303	0.161	0.210	0.317	0.152	0.104	0.185	0.073
DAE [29]	0.251	0.297	0.163	0.111	0.151	0.046	0.224	0.302	0.152	0.206	0.304	0.138	0.104	0.190	0.078
DCGAN [33]	0.265	0.315	0.176	0.120	0.151	0.045	0.210	0.298	0.139	0.225	0.346	0.157	0.121	0.174	0.078
DeCNN [30]	0.240	0.282	0.174	0.092	0.133	0.038	0.227	0.299	0.162	0.186	0.313	0.142	0.098	0.175	0.073
VAE [31]	0.245	0.291	0.167	0.108	0.152	0.040	0.200	0.282	0.146	0.193	0.334	0.168	0.107	0.179	0.079
JULE [3]	0.192	0.272	0.138	0.103	0.137	0.033	0.182	0.277	0.164	0.175	0.300	0.138	0.054	0.138	0.028
DEC [1]	0.257	0.301	0.161	0.136	0.185	0.050	0.276	0.359	0.186	0.282	0.381	0.203	0.122	0.195	0.079
DAC [2]	0.396	0.522	0.306	0.185	0.238	0.088	0.366	0.470	0.257	0.394	0.527	0.302	0.219	0.275	0.111
ADC [32]	-	0.325	-	-	0.160	-	-	0.530	-	-	-	-	-	-	-
DDC [34]	0.424	0.524	0.329	-	-	-	0.371	0.489	0.267	0.433	0.577	0.345	-	-	-
DCCM [35]	0.496	0.623	0.408	0.285	0.327	0.173	0.376	0.482	0.262	0.608	0.710	0.555	0.321	0.383	0.182
IIC [36]	-	0.617	-	-	0.257	-	-	0.610	-	-	-	-	-	-	-
PICA [24]	0.591	0.696	0.512	0.310	0.337	0.171	0.611	0.713	0.531	0.802	0.870	0.761	0.352	0.352	0.201
CC [19]	0.705	0.790	0.637	0.431	0.429	0.266	0.764	0.850	0.726	0.862	0.895	0.825	0.401	0.342	0.225
DeepCluE [7]	0.727	0.764	0.646	0.472	0.457	0.288	-	-	-	0.882	0.924	0.856	0.448	0.416	0.273
DivClust [17]	0.724	0.819	0.681	0.440	0.437	0.283	-	-	-	0.891	0.936	0.878	0.516	0.529	0.376
JDCE (ours)	0.711	0.843	0.688	0.418	0.472	0.325	0.773	0.870	0.747	0.897	0.944	0.902	0.556	0.590	0.403



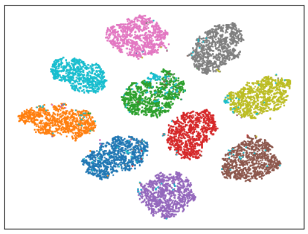
(a) The clustering performance of STL-10 before ensemble.



(b) The clustering performance of STL-10 after ensemble.



(c) The clustering performance of ImageNet-10 before ensemble.



(d) The clustering performance of ImageNet-10 after ensemble.

Fig. 2. t-SNE results on STL-10 and ImageNet-10.

D. Ablation Study

To further evaluate the effects of jointly generating base results and doing the ensemble, in this section, we compare our method with a variant of the two-step way, which is similar to [7] and [17]. Specifically, we first train the base clustering generation module alone and generate multiple base results. Then, we freeze the base clustering generation module and train the ensemble module alone to learn the final consensus result. The results are shown in Table III. We can see that our original model performs much better than the two-step variant, which well demonstrates the superiority of the joint learning framework and is consistent with our motivation.

TABLE III
THE CLUSTERING PERFORMANCE OF DIFFERENT STRATEGIES ON ALL DATASETS. THE BEST SCORE ON EACH DATASETS IS IN **BOLD**.

Datasets	**Two-steps**			**Ours**		
Metrics	***NMI***	***ACC***	***ARI***	***NMI***	***ACC***	***ARI***
CIFAR-10	0.645	0.807	0.599	**0.711**	**0.843**	**0.688**
CIFAR-100	0.366	0.386	0.242	**0.431**	**0.443**	**0.276**
STL-10	0.719	0.824	0.671	**0.766**	**0.861**	**0.738**
ImageNet-10	0.769	0.842	0.756	**0.897**	**0.944**	**0.902**
ImageNet-Dogs	0.458	0.464	0.259	**0.553**	**0.554**	**0.383**

IV. CONCLUSION

In this paper, we propose a new joint deep clustering ensemble framework, which jointly generates the base results with the representation learning and does the ensemble. In this framework, ensemble learning can guide the representation learning and further improve the base results, so that the deep clustering and ensemble can boost each other. We design a base clustering generation module and an ensemble module, and integrate them into a unified framework. The extensive experiments show that our proposed method outperforms the deep clustering methods and even the state-of-the-art deep clustering ensemble methods, which demonstrates the effectiveness and superiority of the proposed method.

ACKNOWLEDGMENTS

This paper is supported by the National Natural Science Foundation of China grants 62176001, and the Natural Science Project of Anhui Provincial Education Department under grant 2023AH030004.

REFERENCES

[1] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *ICML*, pp. 478–487, PMLR, 2016.

[2] J. Chang, L. Wang, G. Meng, S. Xiang, and C. Pan, "Deep adaptive image clustering," in *ICCV*, pp. 5879–5887, 2017.

[3] J. Yang, D. Parikh, and D. Batra, "Joint unsupervised learning of deep representations and image clusters," in *CVPR*, pp. 5147–5156, 2016.

[4] C. Niu, H. Shan, and G. Wang, "Spice: Semantic pseudo-labeling for image clustering," *IEEE TIP*, vol. 31, pp. 7264–7278, 2022.

[5] B. Sun, P. Zhou, L. Du, and X. Li, "Active deep image clustering," *KBS*, vol. 252, p. 109346, 2022.

[6] F. Wang, X. Wang, and T. Li, "Generalized cluster aggregation," in *IJCAI* (C. Boutilier, ed.), pp. 1279–1284, 2009.

[7] D. Huang, D.-H. Chen, X. Chen, C.-D. Wang, and J.-H. Lai, "Deepclue: Enhanced deep clustering via multi-layer ensembles in neural networks," *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2024.

[8] P. Zhou, B. Hu, D. Yan, and L. Du, "Clustering ensemble via diffusion on adaptive multiplex," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 4, pp. 1463–1474, 2024.

[9] A. Strehl and J. Ghosh, "Cluster ensembles — A knowledge reuse framework for combining multiple partitions," *JMLR*, vol. 3, pp. 583–617, 2002.

[10] A. Topchy, A. K. Jain, and W. Punch, "A mixture model for clustering ensembles," in *SDM*, pp. 379–390, SIAM, 2004.

[11] P. Zhou, B. Sun, X. Liu, L. Du, and X. Li, "Active clustering ensemble with self-paced learning," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2023.

[12] N. Iam-On, T. Boongoen, S. Garrett, and C. Price, "A link-based approach to the cluster ensemble problem," *IEEE TPAMI*, vol. 33, no. 12, pp. 2396–2409, 2011.

[13] D. Huang, C.-D. Wang, and J.-H. Lai, "Locally weighted ensemble clustering," *IEEE TCYB*, vol. 48, no. 5, pp. 1460–1473, 2017.

[14] P. Zhou, L. Du, Y.-D. Shen, and X. Li, "Tri-level robust clustering ensemble with multiple graph learning," in *AAAI*, vol. 35, pp. 11125–11133, 2021.

[15] P. Zhou, X. Liu, L. Du, and X. Li, "Self-paced adaptive bipartite graph learning for consensus clustering," *TKDD*, vol. 17, no. 5, pp. 62:1–62:35, 2023.

[16] P. Zhou, L. Du, and X. Li, "Adaptive consensus clustering for multiple k-means via base results refining," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 10, pp. 10251–10264, 2023.

[17] I. M. Metaxas, G. Tzimiropoulos, and I. Patras, "Divclust: Controlling diversity in deep clustering," in *CVPR*, pp. 3418–3428, 2023.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, pp. 770–778, 2016.

[19] Y. Li, P. Hu, Z. Liu, D. Peng, J. T. Zhou, and X. Peng, "Contrastive clustering," in *AAAI*, vol. 35, pp. 8547–8555, 2021.

[20] P. Zhou, L. Du, X. Liu, Z. Ling, X. Ji, X. Li, and Y. Shen, "Partial clustering ensemble," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 5, pp. 2096–2109, 2024.

[21] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, pp. 281–297, Oakland, CA, USA, 1967.

[22] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[23] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *AISTATS*, pp. 215–223, 2011.

[24] J. Huang, S. Gong, and X. Zhu, "Deep semantic clustering by partition confidence maximisation," in *CVPR*, pp. 8849–8858, 2020.

[25] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," *Advances in neural information processing systems*, vol. 14, 2001.

[26] K. C. Gowda and G. Krishna, "Agglomerative clustering using the concept of mutual nearest neighbourhood," *Pattern recognition*, vol. 10, no. 2, pp. 105–112, 1978.

[27] D. Cai, X. He, X. Wang, H. Bao, and J. Han, "Locality preserving nonnegative matrix factorization," in *IJCAI*, 2009.

[28] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," *NeurIPS*, vol. 19, 2006.

[29] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.," *JMLR*, vol. 11, no. 12, 2010.

[30] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," in *CVPR*, pp. 2528–2535, IEEE, 2010.

[31] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[32] P. Haeusser, J. Plapp, V. Golkov, E. Aljalbout, and D. Cremers, "Associative deep clustering: Training a classification network with no labels," in *Pattern Recognition: 40th German Conference, GCPR 2018, Stuttgart, Germany, October 9-12, 2018, Proceedings 40*, pp. 18–32, Springer, 2019.

[33] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[34] J. Chang, Y. Guo, L. Wang, G. Meng, S. Xiang, and C. Pan, "Deep discriminative clustering analysis," *arXiv preprint arXiv:1905.01681*, 2019.

[35] J. Wu, K. Long, F. Wang, C. Qian, C. Li, Z. Lin, and H. Zha, "Deep comprehensive correlation mining for image clustering," in *ICCV*, pp. 8150–8159, 2019.

[36] X. Ji, J. F. Henriques, and A. Vedaldi, "Invariant information clustering for unsupervised image classification and segmentation," in *ICCV*, pp. 9865–9874, 2019.