





Unsupervised Few-Shot Image Classification by Learning Features into Clustering Space

Shuo Li^{1,2,3,4} , Fang Liu^{1,2,3,4}  , Zehua Hao^{1,2,3,4}, Kaibo Zhao^{1,2,3,4},
and Licheng Jiao^{1,2,3,4} 

¹ Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, Xi'an, China

² International Research Center for Intelligent Perception and Computation, Xi'an, China

³ Joint International Research Laboratory of Intelligent Perception and Computation, Xi'an, China

⁴ School of Artificial Intelligent, Xidian University, Xi'an 710071, P.R. China
{alisure,zhao_1995}@stu.xidian.edu.cn, f63liu@163.com,
lchjiao@mail.xidian.edu.cn

Abstract. Most few-shot image classification methods are trained based on tasks. Usually, tasks are built on base classes with a large number of labeled images, which consumes large effort. Unsupervised few-shot image classification methods do not need labeled images, because they require tasks to be built on unlabeled images. In order to efficiently build tasks with unlabeled images, we propose a novel single-stage clustering method: Learning Features into Clustering Space (LF2CS), which first set a separable clustering space by fixing the clustering centers and then use a learnable model to learn features into the clustering space. Based on our LF2CS, we put forward an image sampling and c-way k-shot task building method. With this, we propose a novel unsupervised few-shot image classification method, which jointly learns the learnable model, clustering and few-shot image classification. Experiments and visualization show that our LF2CS has a strong ability to generalize to the novel categories. From the perspective of image sampling, we implement four baselines according to how to build tasks. We conduct experiments on the Omniglot, miniImageNet, tieredImageNet and CIFARFS datasets based on the Conv-4 and ResNet-12 backbones. Experimental results show that ours outperform the state-of-the-art methods.

Keywords: Few-shot learning · Unsupervised few-shot image classification · Single-stage clustering · Learning features into clustering space

1 Introduction

Few-Shot Learning (FSL) aims to learn the novel categories by a small number of images, and usually includes an auxiliary dataset for training [41–43]. The

purpose of image classification is to predict the category of image x , while few-shot image classification predicts which of $c \times k$ images (c categories and each category has k images) are of the same category as image x . Few-shot image classification usually organizes images into a c -way k -shot form, which is called a task. FSL learns meta-knowledge related to tasks, rather than knowledge related to specific categories, thereby generalizing the meta-knowledge to the novel categories. The auxiliary dataset usually contains a large number of annotated images. For example, the training set of tieredImageNet [38] contains 448,695 images. When FSL methods are directly applied to a new field, we need to label the auxiliary dataset, which will consume large effort. To alleviate the burden caused by the auxiliary dataset, Unsupervised FSL (UFSL) has attracted attention [2, 6, 16, 17, 20, 21, 25, 36]. Under the setting of supervised FSL, the labels of the auxiliary dataset are used to build tasks (as shown in Fig. 1 (a)), while under the setting of UFSL, the auxiliary dataset has no available labels. Our work will explore how to build tasks through unlabeled auxiliary dataset.

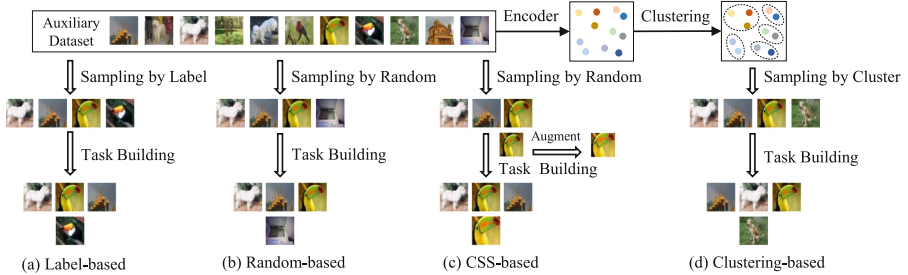


Fig. 1. The four baselines from the view of sampling. (a) Label-based baseline, which is a supervised baseline. Since the images have category labels, two of the four sampled images belong to the same category. (b) Random-based baseline, in which four images are randomly sampled, and the label of task is randomly determined. (c) CSS-based baseline, in which three images are randomly sampled, and then one of the images is selected to obtain another view through data augmentation. (d) Clustering-based baseline, in which first all images are divided into multiple clusters by a clustering algorithm, and then four images are selected with cluster ids as labels.

There are two common unsupervised ways to build tasks from the auxiliary dataset: 1) CSS-based methods (Comparative Self-Supervised, as shown in Fig. 1(c)) use data augmentations to obtain another view of the images to construct the image pairs, and then use the image pairs to build tasks [17, 20]; 2) Clustering-based methods (as shown in Fig. 1(d)), by a clustering algorithm, divide the images into clusters to obtain the pseudo-labels, and then use the pseudo-labels to build tasks [5, 16]. Since CSS-based methods build tasks based on the different views, these methods are concise and effective, but the diversity of tasks may be poor. Also, since Clustering-based methods build tasks based on the clusters, the diversity of tasks may be good, but these methods usually

contain multiple steps. For example, CACTUs [16] use the existing unsupervised model [7] to extract features, and then cluster these features to obtain the cluster ids (cluster assignments). This method takes features learning and clustering as two independent steps and requires additional unsupervised models.

To combine the advantages of both methods, we propose a Learn Features into Clustering Space (LF2CS) method, which is a single-stage clustering method. By fixing the clustering center matrix to the identity matrix, LF2CS sets a separable clustering space, and then uses a learnable model to learn features into the clustering space. By this way, LF2CS does not require any other images when determining the cluster id of each image. Based on LF2CS, we put forward an image sampling and task building method, and thus propose a clustering-based unsupervised few-shot image classification method, as shown in Fig. 2. First, each image x_i is fed into a random initialized network to obtain the feature z_i , and LF2CS is used to obtain the initial cluster id \hat{y}_i , so that the cluster ids \hat{Y} of all images can be initialized. Then, for each image x_i , $c \times k$ images are sampled based on the cluster ids \hat{Y} , in which k images have the same cluster id as the image x_i , so a task can be built based on these $c \times k + 1$ images. Finally, encoders are used to extract features of the images in the task, and the new cluster id \hat{y}_i of the image x_i is calculated by LF2CS and updated to the cluster ids \hat{Y} .

Our method first generate cluster ids of images, then samples images to build tasks, and realizes joint learning of feature extraction, clustering and FSL. After the task is built, any task-based few-shot learning method [41–43] can be used to optimize the network, such as MatchNet [43]. To show the performance of our method, as shown in Fig. 1, we summarize four baselines according to how to build tasks from the perspective of image sampling. Based on the Conv-4 [43] and ResNet-12 backbones, we conduct experiments on the Omniglot [23], miniImageNet [43], tieredImageNet [38], and CIFARFS [3] datasets. In summary, our main contributions are as follows:

- 1) By fixing the clustering center matrix to the identity matrix, we first set a separable clustering space and then use a learnable model to learn features into the clustering space;
- 2) A clustering-based unsupervised few-shot image classification method is proposed through image sampling and task building, which jointly learns feature learning, clustering, and few-shot image classification;
- 3) From the view of sampling, we implement four baselines according to how to build tasks: Label-based, Random-based, CSS-based, and Clustering-based;
- 4) We conduct experiments on a series of benchmarks based on Conv-4 and ResNet-12, and experimentals show that our method has a strong ability to generalize to the novel categories and achieves the state-of-the-art results.

2 Related Work

2.1 Few-Shot Learning

FSL aims to learn the novel categories through few labeled images, and it usually contains a labeled auxiliary dataset [12, 26, 30, 32]. There are many methods: Metric-based, Finetune-based, and Parameterize-based, etc. Metric-based methods are to learn the ability of similarity measure [28, 41–43]. Finetune-based methods are to finetune a base-learner for new tasks by its few samples and make the base-learner converge fast within fewer parameter update steps [11, 37, 40]. Parameterize-based methods are to parameterize the base-learner or its sub-components, thereby generalizing the meta-learner to new tasks [4, 8, 44].

2.2 Unsupervised Feature Learning

Unsupervised feature learning includes many methods, and we mainly introduce clustering and self-supervised learning. Clustering aims to divide the data into different clusters based on the inherent information of data, such as Gaussian Mixture model [1], K-means [18], spectral clustering [48], hierarchical clustering [19], etc. Recently, deep-based clustering methods [7, 27, 45, 47] have shown superior performance. Self-supervised learning uses the data itself to construct some supervised signals to replace the label, such as colorization [46], solving jigsaw puzzle [34], rotation [13], etc. Recently, comparative self-supervised learning methods [9, 10, 14, 15, 31] have made great progress, which learn from multiple views of the images. These methods obtain multiple different views, thereby constructing positive and negative image pairs, so that the positive image pair are similar in the feature space, and the negative image pair are not similar [15].

2.3 Unsupervised Few-Shot Learning

Different from FSL, UFSL usually contains an unlabeled auxiliary dataset. There are two common methods: CSS-based and Clustering-based. CSS-based methods use data augmentations to obtain mutiply views of image to construct image pairs which are used to build tasks. Generally, the data augmentations contain common image transforms [2, 6, 17, 20, 25, 36], such as random resized crop and color jitter, and (or) generative models [21, 36], such as autoencoders and generative adversarial networks. Clustering-based methods [5, 16] divide the images in the auxiliary dataset into multiple clusters to obtain the pseudo-labels of the images, and then use the pseudo-labels to build tasks.

3 Our Approach

3.1 Unsupervised Few-Shot Learning

In FSL, an auxiliary dataset $D_{aux} = \{(x_i, y_i)\}_{i=1}^{N_{aux}}$ (N_{aux} is the number of images) is used to train a learnable model, and then applies the model to a

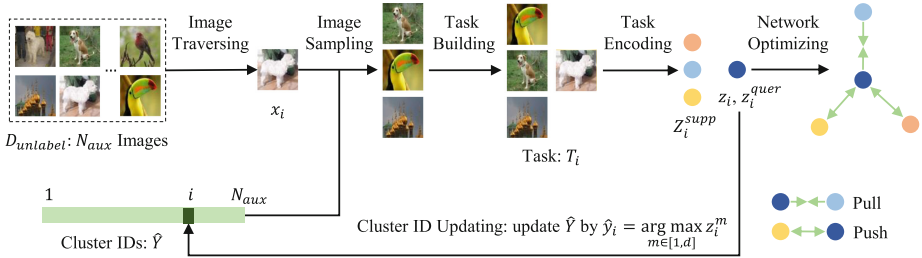


Fig. 2. The overall pipeline of our approach. The 3-way 1-shot task is used as an example to show our method. Clusters IDs \hat{Y} stores the cluster ids of all images in $D_{unlabel}$, and $|\hat{Y}| = N_{aux}$. First, for the image x_i , its cluster id \hat{y}_i is obtained from \hat{Y} (Image Traversing). Then, according to \hat{y}_i and \hat{Y} , three images are sampled from $D_{unlabel}$: the cluster id of one image is \hat{y}_i , and the cluster id of the other two images is not \hat{y}_i (Image Sampling), so the 3-way 1-shot task can be built (Task Building). Then, the feature of task is obtained by encoders (Task Encoding), and encoders are optimized by reducing the distance of features with the same cluster id as the image x_i (Pull) and increasing the distance of features with different cluster ids to the image x_i (Push) (Network Optimizing). Finally, the cluster id \hat{y}_i is updated (Cluster ID Updating).

target dataset $D_{target} = \{\{(x_i, y_i)\}_{i=1}^{N_{label}}, \{(x_j, y_j)\}_{j=1}^{N_{unlabel}}\}$ (where $N_{label} \ll N_{unlabel}$, N_{label} and $N_{unlabel}$ are the number of labeled and unlabeled images), so that the target dataset with few labeled images can be classified. D_{aux} and D_{target} are also known as the base classes and the novel classes, respectively. D_{aux} contains a large number of labeled images. UFSL aims to train a model using an unlabeled auxiliary dataset $D_{unlabel} = \{x_i\}_{i=1}^{N_{aux}}$ which contains a large number of unlabeled images, and then also applies the model to D_{target} .

Task-based FSL organizes images into tasks in the form of c -way k -shot. Task T_i is composed of the support set T_i^{supp} , the query set T_i^{quer} and the label y_i^{task} :

$$T_i = \{T_i^{supp}, T_i^{quer}, y_i^{task}\} = \{\{(x_j, y_j)\}_{j=1}^{c \times k}, \{x_i\}, y_i^{task}\}, \quad (1)$$

where $y_i^{task} \in \{0, 1\}^{c \times k}$, T_i^{supp} contains c classes and each class has k labeled images. Note that in our work, T_i^{quer} contains only one image x_i . Under the setting of supervised FSL, since the image label is available on the base classes, the task T_i is built in a supervised way. Under the setting of UFSL, the task T_i needs to be built in an unsupervised way. The difference between most UFSL methods is how to build tasks in the base classes. There are two common unsupervised ways to build tasks: Clustering-based and CSS-based. Clustering-based methods divide the images of $D_{unlabel}$ into multiple clusters, and then uses the cluster ids as the pseudo-labels to build tasks. CSS-based methods build tasks based on different views of the same image.

3.2 Learn Features into Clustering Space

In DeepCluster [7], Caron *et. al* alternates between clustering the image features to generate pseudo-labels (Eq. 2) and updating the parameters of the learnable

model f_θ by predicting pseudo-labels (Eq. 3):

$$\min_{C \in \mathbb{R}^{d \times K}} \frac{1}{N} \sum_{i=1}^N \min_{\hat{y}_i \in \{0,1\}^K} \|f_\theta(x_i) - C\hat{y}_i\|_2^2, \quad (2)$$

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^N \mathcal{L}(f_\theta(x_i), \hat{y}_i), \quad (3)$$

where θ is the parameter of the network $f_\theta(\cdot)$, C is the cluster center matrix, K is the number of clusters, d is the output feature dimension of $f_\theta(\cdot)$, and \hat{y}_i is the cluster id of the image x_i and $\hat{y}_i^T \mathbf{1}_K = 1$. This method has some limitations: all features need to be extracted before clustering, feature learning and clustering are two independent step, and clustering result usually depends on the initialization of the cluster centers.

Different from [7], we preset a separable d -dimensional clustering space by setting $K = d$ and fixing the matrix C to the identity matrix $E \in \mathbb{R}^{d \times d}$. Since it is located on the coordinate axis of this space, all cluster centers are orthogonal, this clustering space has strong separability. Therefore, Eq. 2 is simplified to:

$$\frac{1}{N} \sum_{i=1}^N \min_{\hat{y}_i \in \{0,1\}^d} \|f_\theta(x_i) - E\hat{y}_i\|_2^2. \quad (4)$$

In order to make Eq. 4 feasible, the network $f_\theta(\cdot)$ need to be used to learn features of all images into the d -dimensional clustering space. We call this Learn Features into Clustering Space (LF2CS). Specifically, given an image x_i , the feature $z_i = f_\theta(x_i)$ and the cluster to which x_i belongs can be quickly calculated by:

$$\hat{y}_i = \arg \min_{m \in [1,d]} \|z_i - E_m\|_2^2 = \arg \max_{m \in [1,d]} z_i^m, \quad (5)$$

where $z_i \in \mathbb{R}^d$, $E_m \in \mathbb{R}^d$, E_m is the cluster center of cluster m , z_i^m is the m -th value of z_i , and \hat{y}_i is the cluster id of x_i . By treating \hat{y}_i as the pseudo-label of x_i , the parameter of $f_\theta(x_i)$ can be updated by Eq. 3.

From Eq. 5, it can be seen that given a feature z_i , cluster center, which is closest to z_i , can be quickly calculated. The advantage of this is that the cluster id \hat{y}_i of the image x_i can be calculated in a simple way, without the need of other image features. Our LF2CS eliminates the process of solving cluster centers in Eq. 2 by iterative algorithms, such as K -means. Therefore, our LF2CS is a single-stage clustering algorithm, instead of alternating between Eq. 3 and Eq. 2.

Avoiding Trivial Solution. Most clustering-based methods have trivial solutions [7, 27]. In order to avoid invalid solutions, our LF2CS forces all images to be scattered into the clustering space by limiting the maximum number of images in each cluster. When the number of images in the m -th cluster exceeds $n_{max} = N_{aux}/d$, we will no longer assign any images to the m -th cluster, but look for the next cluster that satisfies the constraint. The maximum number of

images n_{max} per cluster means that each cluster is almost the same size. In this way, our method avoids both over-clustering (many clusters are relatively small) and under-clustering (some clusters are relatively large).

3.3 Image Sampling and Task Building

Given an unlabeled auxiliary dataset $D_{unlabel} = \{x_i\}_{i=1}^{N_{aux}}$, our goal is to build tasks from unlabeled images. In order to record the cluster to which each image belongs, $\hat{Y} = \{\hat{y}_i\}_{i=1}^{N_{aux}}$ is used to store the cluster ids of all images, that is, \hat{y}_i is the cluster id of the image x_i , where $\hat{y}_i \in [1, d]$. To facilitate joint training, we use mini-batch of tasks in training for each optimization step, instead of episode

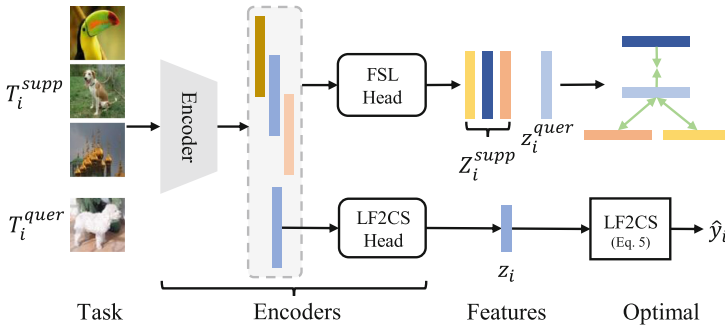


Fig. 3. Task Encoding and Network Optimizing. Both FSL Head and LF2CS Head are a fully connected layer. We call Encoder, FSL Head, and LF2CS Head as Encoders. We treat the optimization as a multi-task learning problem: LF2CS and few-shot image classification. Note that $z_i^{quer} \in \mathbb{R}^q$ and $z_i \in \mathbb{R}^d$.

[43]. When evaluating, like other methods [41–43], we use episode as input. To build c -way k -shot task, the image x_i is used as the query set T_i^{quer} , $c \times k$ images are selected as the support set T_i^{supp} through the cluster id \hat{y}_i . Specifically, we get c clusters by \hat{y}_i and randomly selecting other $c - 1$ clusters:

$$\hat{Y}_c = \{\hat{y}_i\} \cup \{c_j\}_{j=1}^{c-1}, \quad s.t. \quad c_j \neq \hat{y}_i, \quad (6)$$

where c_j is a cluster id and \hat{Y}_c is the set of c cluster ids. According to \hat{Y}_c , by randomly selecting k images from $D_{unlabel}$ for each cluster id in \hat{Y}_c , we can get $c \times k$ images which are treated as the support set T_i^{supp} . Since the images with the same cluster id are usually treated as the same category, the task $T_i = \{T_i^{supp}, T_i^{quer}, y_i^{task}\}$ is built successfully, where y_i^{task} can be obtained by whether the cluster ids of the images in the query set T_i^{quer} and the support set T_i^{supp} are the same. The new cluster id \hat{y}_i of the image x_i can be calculated by Eq. 5, and then \hat{Y} is updated by replacing the old cluster id with the new one.

3.4 Task Encoding and Network Optimizing

As shown in Fig. 3, we use the Encoder to embed images of the task T_i . In our method, the Encoder is Conv-4 or ResNet-12. FSL Head is used to learn features for few-shot image classification, and LF2CS Head is used to learn features into the clustering space. Both FSL Head and LF2CS Head are a fully connected layer, which are used to embed features for clustering and few-shot image classification. The output dimension of FSL Head is q and the output dimension of LF2CS Head is d . The feature $z_i \in \mathbb{R}^d$ is extracted by feeding the image x_i in the query set T_i^{quer} to Encoder and LF2CS Head, and the feature $z_i^{quer} \in \mathbb{R}^q$ also is extracted by feeding the image x_i in the query set T_i^{quer} to Encoder and FSL Head. In addition, the features $Z_i^{supp} = \{z_{i,j}^{supp}\}_{j=1}^{c \times k}$ are extracted by feeding images in the support set T_i^{supp} to Encoder and FSL Head, where $z_{i,j}^{supp} \in \mathbb{R}^q$ is the feature of the j -th image in the support set T_i^{supp} .

Algorithm 1 The overall pipeline of our method.

Input: the dataset $D_{unlabel} = \{x_i\}_{i=1}^{N_{aux}}$, max iteration T_{max} ;

Output: Encoders.

- 1: Initialize the parameters of Encoders;
 - 2: **for** $i = \{1, 2, \dots, N_{aux}\}$ **do**
 - 3: Extract z_i and compute \hat{y}_i ; // Initialize \hat{Y} ;
 - 4: **end for**
 - 5: **for** $t = \{1, 2, \dots, T_{max}\}$ **do**
 - 6: // Image Traversing
 - 7: **for** $i = \{1, 2, \dots, N_{aux}\}$ **do**
 - 8: Get \hat{Y}_c by \hat{y}_i and Select $c \times k$ images as T_i^{supp} ; // Image Sampling
 - 9: Build task $T_i = \{T_i^{supp}, T_i^{quer}, y_i^{task}\}$; // Task Building
 - 10: Get features $z_i, z_i^{quer}, Z_i^{supp}$ by Encoders; // Task Encoding
 - 11: Compute loss \mathcal{L} by Eq.8 and Update the parameters; // Network Optimizing
 - 12: Calculate the cluster id \hat{y}_i and Update \hat{Y} by \hat{y}_i ; // Cluster ID Updating
 - 13: **end for**
 - 14: **end for**
 - 15: **return** Encoders;
-

After the task is built, any task-based few-shot learning method [41–43] can be used to optimize the network. We optimize the network based on MatchNet [43] and use the softmax over cosine to measure similarity between features:

$$p_i^{task} = \{p_{i,j}^{task}\}_{j=1}^{c \times k} = \left\{ \frac{\exp(\cos(z_{i,j}^{supp}, z_i^{quer}))}{\sum_{z_{i,l}^{supp} \in Z_i^{supp}} \exp(\cos(z_{i,l}^{supp}, z_i^{quer}))} \right\}_{j=1}^{c \times k}, \quad (7)$$

where $\cos(\cdot, \cdot)$ is the cosine similarity. FSL can be optimized by MSE (Mean Squared Error) loss between p_i^{task} and the task label y_i^{task} , and LF2CS can be optimized by CE (Cross Entropy) loss between the feature z_i and the cluster id \hat{y}_i . The optimization of our method can be treated as a multi-task learning problem. Therefore, we jointly learn LF2CS and FSL, and the total loss is:

$$\mathcal{L} = MSE(p_i^{task}, y_i^{task}) + CE(z_i, \hat{y}_i), \quad (8)$$

where $y_i^{task} \in \{0, 1\}^{c \times k}$, $p_i^{task} \in \mathbb{R}^{c \times k}$, $z_i \in \mathbb{R}^d$, and $\hat{y}_i \in [1, d]$. In the CE loss, \hat{y}_i is converted to a one-hot vector. Taking Eq. 8 as our total loss, our method increases the similarity of features with the same cluster id and reduces the similarity of features with different cluster ids.

For easy understanding, we show the overall pipeline of our method in Algorithm 1. Please refer to <https://github.com/xidianai/LF2CS> for the code.

4 Experimental Results

4.1 Setup

Our method includes an Encoder and two linear head (FSL Head and LF2CS Head). For Encoder, we use two widely used networks (Conv-4 [43] and ResNet-12). For each Head, we use a fully connected layer. The output dimension q of FSL Head is 1024, and the output dimension d of LF2CS Head is 1024. All our models are trained from scratch without any pre-trained weights. We use the SGD optimizer with learning rate of 0.01, weight decay of $5e-4$ and batch size of 64. We use the learning rate strategy shown in Fig. 5 and train the model for 1,500 epochs. All our experiments are implemented based on PyTorch [35]. We use the evaluate strategies used in [42]. Following [42], each class uses 15 query samples in each episode: for c -way k -shot, each episode contains $15 \times c$ query images and $c \times k$ support images. All accuracies are averaged over 1,000

Table 1. Few-shot image classification accuracies of 5-way and 20-way on Omniglot. All accuracies are averaged over 1000 test episodes and are reported with 95% confidence intervals. The backbone of all methods is Conv-4. Bold represent the best values.

	Method	5-Way Acc.		20-Way Acc.	
		1-Shot	5-Shot	1-Shot	5-Shot
FSL	MatchNet [43]	98.10±-%	98.90±-%	93.80±-%	98.50±-%
	MAML [11]	98.70±0.40%	99.90±0.10%	95.80±0.30%	98.90±0.20%
	ProtoNet [41]	98.80±-%	99.70±-%	96.00±-%	98.90±-%
	Meta-SGD [29]	99.50±-%	99.90±-%	95.90±-%	99.00±-%
	RelationNet [42]	99.60±0.20%	99.80±0.10%	97.60±0.20%	99.10±0.10%
	Baselines: Label	97.92±0.22%	99.47±0.08%	92.84±0.21%	97.68±0.10%
UFSL	CACTUs [16]	68.84±-%	87.78±-%	48.09±-%	73.36±-%
	UMTRA [20]	83.80±-%	95.43±-%	74.25±-%	92.12±-%
	LASIUM [21]	83.26±0.55%	95.29±0.22%	-	-
	ProtoTransfer [33]	88.00±-%	96.48±-%	72.27±-%	89.08±-%
	AAL [2]	88.40±0.75%	98.00±0.32%	70.20±0.86%	88.30±1.22%
	ULDA [36]	91.00±0.42%	98.14±0.15%	78.05±0.31%	94.08±0.13%
	UFLST [17]	97.03±-%	99.19±-%	91.28±-%	97.37±-%
	Baselines: Random	58.50±0.70%	71.73±0.59%	33.94±0.31%	47.14±0.32%
	Baselines: CSS	83.97±0.56%	94.65±0.26%	65.25±0.34%	84.74±0.22%
	Baselines: Clustering	83.14±0.62%	91.67±0.36%	61.52±0.36%	77.05±0.28%
Ours: LF2CS	97.31±0.25%	99.32±0.10%	91.72±0.22%	97.65±0.09%	

test episodes and are reported with 95% confidence intervals. We also use KNN to evaluate our LF2CS, and set the number of nearest neighbor samples to 100. The top-1 and top-5 accuracy of KNN are used to report the results.

4.2 Unsupervised Few-Shot Image Classification Results

Results on Omniglot. The Omniglot [23] dataset consists of 1,623 characters from 50 different alphabets and each character contains 20 samples drawn by different people. Following [41, 43], we resize the sample to 28×28 . We use 1,028 characters for training, 172 characters for validation, and the remaining 423 characters for testing. On the Omniglot dataset, we only use Conv-4 as the backbone of Encoder. Table 1 shows the few-shot image classification accuracies on Omniglot. Compared with other unsupervised baselines, the method of building tasks by randomly selecting images (Random-based) has the worst accuracies and our LF2CS has the best accuracies in all cases. Compared with other unsupervised methods, ours do 97.31% for 5-way 1-shot with Conv-4 as the backbone, which has also reached the level of supervised methods.

Table 2. 5-way 1-shot and 5-way 5-shot few-shot image classification accuracies on the miniImageNet dataset. All accuracies are averaged over 1000 test episodes and reported with 95% confidence intervals. Bold represent the best values.

	Method	Backbone	5-Way 1-Shot	5-Way 5-Shot
FSL	MatchNet [43]	Conv-4	46.60±-%	60.00±-%
	ProtoNet [41]	Conv-4	49.42±0.78%	68.20±0.66%
	RelationNet [42]	Conv-4	50.44±0.82%	65.32±0.70%
	MAML [11]	Conv-4	48.70±1.84%	63.11±0.92%
	MetaOptnet [24]	ResNet-12	62.64±0.61%	78.63±0.46%
	Baselines: Label	Conv-4	52.53±0.62%	65.98±0.53%
	Baselines: Label	ResNet-12	60.06±0.69%	71.76±0.58%
	UFSL	UFLST [17]	Conv-4	33.77±0.70%
AAL [2]		Conv-4	37.67±0.39%	40.29±0.68%
CACTUs [16]		Conv-4	39.90±0.74%	53.97±0.70%
UMTRA [20]		Conv-4	39.93±-%	50.73±-%
LASIUM [21]		Conv-4	40.19±0.58%	54.56±0.55%
ULDA [36]		Conv-4	40.63±0.61%	56.18±0.59%
CSSL-FSL [25]		ResNet-50	48.53±1.26%	63.13±0.87%
No-Labels [6]		ResNet-50	50.10±0.20%	60.10±0.20%
Baselines: Random		Conv-4	25.29±0.39%	28.86±0.39%
Baselines: CSS		Conv-4	41.00±0.56%	52.17±0.54%
Baselines: Clustering		Conv-4	41.01±0.59%	51.52±0.55%
Ours: LF2CS		Conv-4	48.32±0.64%	61.52±0.52%
Baselines: Random		ResNet-12	29.92±0.46%	36.82±0.48%
Baselines: CSS		ResNet-12	45.42±0.60%	58.37±0.54%
Baselines: Clustering		ResNet-12	48.48±0.63%	61.10±0.58%
Ours: LF2CS		ResNet-12	53.14±0.62%	67.36±0.50%

Results on MiniImageNet. The miniImageNet [43] dataset consists of 100 categories sampled from ImageNet [39], all images are 84×84 in size, and each category has 600 images. Following [43], we divide all categories into 64 (training), 16 (validation), and 20 (test) categories. Table 2 shows the few-shot image classification accuracies on miniImageNet. We compare with 5 supervised methods and 8 unsupervised methods. Compared with other unsupervised baselines, the method of building tasks by randomly selecting images has the worst accuracies, while our methods have the best accuracies. Especially, our ResNet-12 reaches 53.14% for 5-way 1-shot, which has better results than all unsupervised methods. Compared with other unsupervised methods, our methods have a smaller gap with the supervised few-shot image classification methods.

Table 3. 5-way 1-shot and 5-way 5-shot few-shot image classification accuracies on the tieredImageNet dataset. All accuracies are averaged over 1000 test episodes and reported with 95% confidence intervals. Bold represent the best values.

	Method	Backbone	5-Way 1-Shot	5-Way 5-Shot
FSL	ProtoNet [41]	Conv-4	53.31±0.89%	72.69±0.74%
	RelationNet [42]	Conv-4	54.48±0.93%	71.32±0.78%
	MAML [11]	Conv-4	51.67±1.81%	70.30±1.75%
	MetaOptnet [24]	ResNet-12	65.99±0.72%	81.56±0.53%
	Baselines: Label	Conv-4	56.09±0.72%	68.86±0.60%
	Baselines: Label	ResNet-12	64.67±0.76%	76.71±0.57%
UFSL	ULDA [36]	Conv-4	41.77±0.65%	56.78±0.63%
	Baselines: Random	Conv-4	24.81±0.36%	28.80±0.40%
	Baselines: CSS	Conv-4	40.74±0.59%	52.72±0.58%
	Baselines: Clustering	Conv-4	42.60±0.62%	55.11±0.57%
	Ours: LF2CS	Conv-4	49.15±0.65%	62.54±0.58%
	Baselines: Random	ResNet-12	31.58±0.50%	38.82±0.53%
	Baselines: CSS	ResNet-12	43.13±0.62%	56.36±0.56%
	Baselines: Clustering	ResNet-12	44.93±0.64%	57.53±0.59%
	Ours: LF2CS	ResNet-12	53.16±0.66%	66.59±0.57%

Results on TieredImageNet. The tieredImageNet [38] dataset consists of 608 classes with a total of 779,165 images of size 84×84 . These classes are selected from 34 higher-level nodes in the ImageNet hierarchy. Following [38], 351 classes from 20 high level nodes are used for training, 97 classes from 6 nodes for validation and 160 classes from 8 nodes for testing. Table 3 shows the few-shot image classification accuracies on tieredImageNet. Compared with other unsupervised methods, our methods have the best accuracies. Among them, our ResNet-12 has an accuracy of 53.16% for 5-way 1-shot and 66.59% for 5-way 5-shot. Compared with other unsupervised baselines, our methods have a smaller gap with the supervised few-shot image classification methods.

Results on CIFARFS. The CIFARFS [3] dataset consists of 100 classes sampled from CIFAR100 [22]. All the images are 32×32 in size. Following [3], we divide all classes into 64, 16, and 20 classes for training, validation, and testing, respectively. We compare with four supervised methods and one unsupervised method. Table 4 shows the few-shot accuracies on CIFARFS. Compared with other unsupervised methods, our methods have the best accuracies. For 5-way 1-shot, our method based on the Conv-4 backbone achieves an accuracy of 51.52%.

4.3 Ablation Experiments and Visualization

The Performance of LF2CS. The number of clusters is the same as the dimension d , which usually affects the clustering results. We conduct experiments to observe the effect of the dimension d . Encoder also plays an important role on the clustering results, and we evaluate our LF2CS on two architectures: Conv-4 [43] and ResNet-12. Figure 4 shows the top-1 KNN accuracies and training time per epoch on miniImageNet. Table 5 shows the accuracies of LF2CS with different Encoder on miniImageNet. We report the results with top-1 and top-5 KNN accuracies, and 5-way 1-shot and 5-way 5-shot few-shot accuracies. Note that the categories in training, validation and test sets do not intersect. From Fig. 4, we can find that as the dimension d increases, the KNN accuracy and training time also increase. To balance the time and accuracy, the dimension d

Table 4. Few-shot image classification accuracies on the CIFARFS dataset. All accuracies are averaged over 1000 test episodes and reported with 95% confidence intervals. Bold represent the best values.

	Method	Backbone	5-way 1-shot	5-way 5-shot
FSL	ProtoNet	Conv-4	55.50±0.70%	72.60±0.60%
	RelationNet	Conv-4	55.00±1.00%	69.30±0.80%
	MAML	Conv-4	58.90±1.90%	71.50±1.00%
	MetaOptnet	ResNet-12	72.08±0.70%	85.00±0.50%
	Baselines: Label-based	Conv-4	65.65±0.74%	76.75±0.57%
	Baselines: Label-based	ResNet-12	67.14±0.76%	77.46±0.54%
UFSL	No-Labels	ResNet-50	53.00±0.20%	62.50±0.20%
	Baselines: Random-based	Conv-4	30.87±0.47%	38.03±0.49%
	Baselines: CSS-based	Conv-4	42.59±0.65%	55.64±0.59%
	Baselines: Clustering-based	Conv-4	44.72±0.66%	58.21±0.57%
	Ours: LF2CS	Conv-4	51.52±0.72%	66.82±0.57%
	Baselines: Random-based	ResNet-12	34.25±0.55%	44.48±0.55%
	Baselines: CSS-based	ResNet-12	46.46±0.67%	61.39±0.59%
	Baselines: Clustering-based	ResNet-12	44.88±0.67%	58.50±0.59%
	Ours: LF2CS	ResNet-12	55.04±0.72%	70.62±0.57%

is set to 1024. From the few-shot accuracies in Table 5, it can be seen that LF2CS has a strong generalization ability to the novel categories, such as 48.41% in the training set v.s. 48.32% in the test set with Conv-4 as the backbone and 53.59% in the training set v.s. 53.14% in the test set with ResNet-12 as the backbone.

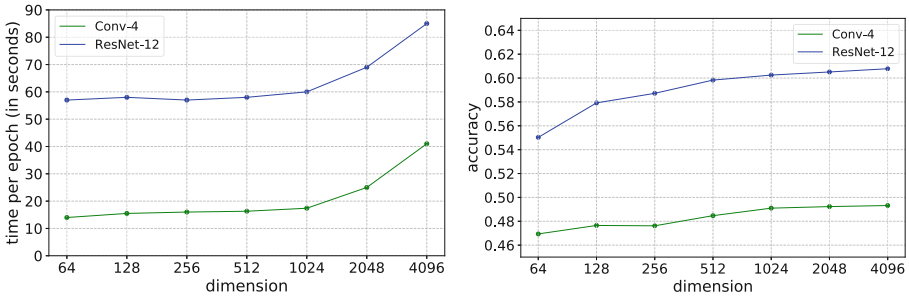


Fig. 4. Left: The Top-1 KNN accuracies in test set on miniImageNet; Right: The training time per epoch of different dimension of LF2CS Head on miniImageNet.

Table 5. Top-1 and Top-5 KNN accuracies, and 5-way 1-shot and 5-way 5-shot few-shot accuracies of LF2CS Head with different Encoder on the training, validation and test sets of miniImageNet. All few-shot accuracies are averaged over 1000 test episodes and reported with 95% confidence intervals. The dimension d is 1024.

SubSet	Encoder	Top-1	Top-5	5-Way 1-Shot	5-Way 5-Shot
Training	Conv-4	36.85%	69.36%	48.41±0.62%	62.28±0.50%
	ResNet-12	53.42%	81.61%	53.59±0.65%	68.06±0.51%
Validation	Conv-4	50.49%	88.86%	46.71±0.65%	59.32±0.53%
	ResNet-12	62.17%	92.34%	51.69±0.67%	65.41±0.53%
Test	Conv-4	49.10%	88.00%	48.32±0.64%	61.52±0.52%
	ResNet-12	60.25%	91.38%	53.14±0.62%	67.36±0.50%

The Change of the Cluster Assignments. Now, we show the gradual change of the cluster ids over time. During the training process, we have counted the change of the cluster ids. The change of learning rate is shown in the left of Fig. 5. There are two curves in the right of Fig. 5. The first represents the change ratio of cluster ids and the second represents the ratio of cluster ids that do not meet the limit on the number of clusters. At the beginning of training, the change ratio of cluster ids and the ratio of cluster ids that do not meet the limit on the number of clusters are almost 98% or more. It is almost completely random. But

as the training progresses, the ratio of cluster ids that do not meet the limit on the number of clusters dropped sharply. That is to say, the number of images that do not meet the limit is rapidly decreasing. As the training progresses, the change ratio of cluster assignments is also reduced. At the end of training, only 20% of the cluster assignments changed.

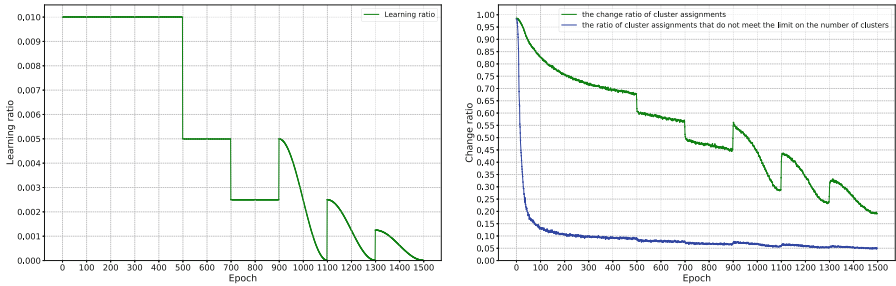


Fig. 5. The gradual change of the cluster ids over time. Left: The learning rate decay using cosine strategy; Right: The change ratio of the cluster ids (assignments).

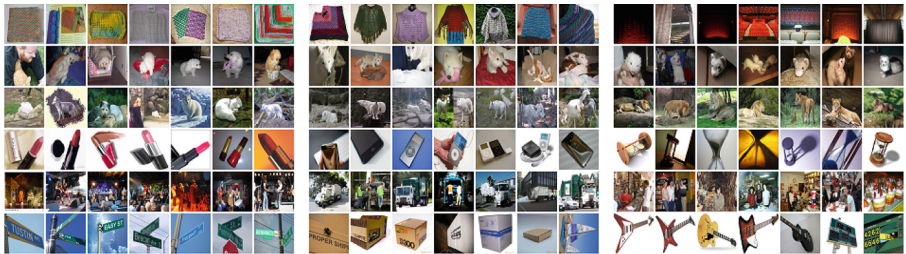


Fig. 6. Visualize images in some clusters on the miniImageNet dataset. Each row of images represents the same clusters. The images of left, middle, and right come from the training, validation and test sets, respectively.

The Visualization of Images in Some Clusters. Since task is built by selecting images from clusters, the clustering results affect the quality of tasks. When there are more images belonging to the same category in the same cluster, the quality of the task will be higher. Therefore, visualizing the images of the same cluster will help to understand our method. Figure 6 shows some images in some clusters, which are divided into three parts. The left, middle, and right show the images in the training, validation, and test sets of miniImageNet, respectively. Obviously, we can see that each cluster represents a certain vision semantics of images. For example, the first row has a similar texture, and the last row has a similar shape.

5 Conclusion

In our work, we propose a novel single-stage clustering method: Learning Features into Clustering Space (LF2CS), which fixes the cluster center matrix to the identity matrix, thereby setting a strongly separable clustering space, and then learns features into the clustering space. Based on this, we put forward an image sampling and task building method, and with this, we propose an unsupervised few-shot image classification method. Experimental results and visualization show that our LF2CS has a strong ability to generalize to the novel categories. Based on Conv-4 and ResNet-12, we conduct experiments on four FSL datasets, and our method achieves the state-of-the-art results.

Acknowledgements. This work was supported in part by the National Natural Science Foundation of China (No.62076192), Key Research and Development Program in Shaanxi Province of China (No.2019ZDLGY03-06), the State Key Program of National Natural Science of China (No.61836009), in part by the Program for Cheung Kong Scholars and Innovative Research Team in University (No. IRT_15R53), in part by The Fund for Foreign Scholars in University Research and Teaching Programs (the 111 Project) (No. B07048), in part by the Key Scientific Technological Innovation Research Project by Ministry of Education, the National Key Research and Development Program of China.

References

1. Abramson, N., Braverman, D.J., Sebestyen, G.S.: Pattern recognition and machine learning. *JASA* **103**(482), 886–887 (2006)
2. Antoniou, A., Storkey, A.J.: Assume, augment and learn: Unsupervised few-shot meta-learning via random labels and data augmentation. *CoRR* abs/1902.09884 (2019)
3. Bertinetto, L., Henriques, J.F., Torr, P.H.S., Vedaldi, A.: Meta-learning with differentiable closed-form solvers. In: *ICLR* (2019)
4. Bertinetto, L., Henriques, J.F., Valmadre, J., Torr, P.H.S., Vedaldi, A.: Learning feed-forward one-shot learners. In: *NIPS*, pp. 523–531 (2016)
5. Bertugli, A., Vincenzi, S., Calderara, S., Passerini, A.: Few-shot unsupervised continual learning through meta-examples. *CoRR* abs/2009.08107 (2020)
6. Bharti, A., Balasubramanian, V.N., Jawahar, C.V.: Few shot learning with no labels. *CoRR* abs/2012.13751 (2020)
7. Caron, M., Bojanowski, P., Joulin, A., Douze, M.: Deep clustering for unsupervised learning of visual features. In: *ECCV*. vol. 11218, pp. 139–156 (2018)
8. Chen, M., et al.: Diversity transfer network for few-shot learning. In: *AAAI*, pp. 10559–10566 (2020)
9. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.E.: A simple framework for contrastive learning of visual representations. In: *ICML*, vol. 119, pp. 1597–1607 (2020)
10. Cui, Y., Liu, F., Liu, X., Li, L., Qian, X.: TCSPANET: two-staged contrastive learning and sub-patch attention based network for polsar image classification. *Remote Sens.* **14**(10), 2451 (2022)
11. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: *ICML*, vol. 70, pp. 1126–1135 (2017)

12. Frikha, A., Krompaß, D., Köpken, H., Tresp, V.: Few-shot one-class classification via meta-learning. In: AAAI, pp. 7448–7456 (2021)
13. Gidaris, S., Singh, P., Komodakis, N.: Unsupervised representation learning by predicting image rotations. In: ICLR (2018)
14. Grill, J., et al.: Bootstrap your own latent - a new approach to self-supervised learning. In: NeurIPS (2020)
15. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.B.: Momentum contrast for unsupervised visual representation learning. In: CVPR, pp. 9726–9735 (2020)
16. Hsu, K., Levine, S., Finn, C.: Unsupervised learning via meta-learning. In: ICLR (2019)
17. Ji, Z., Zou, X., Huang, T., Wu, S.: Unsupervised few-shot feature learning via self-supervised training. *Front. Comput. Neurosci.* **14**, 83 (2020)
18. Jiao, L., Ronghua, S., Fang, L., Weitong, Z.: *Brain and Nature-Inspired Learning, Computation and Recognition*. Elsevier, Amsterdam (2020)
19. Karypis, G., Han, E., Kumar, V.: Chameleon: Hierarchical clustering using dynamic modeling. *Computer* **32**(8), 68–75 (1999)
20. Khodadadeh, S., Bölöni, L., Shah, M.: Unsupervised meta-learning for few-shot image classification. In: NeurIPS, pp. 10132–10142 (2019)
21. Khodadadeh, S., Zehtabian, S., Vahidian, S., Wang, W., Lin, B., Bölöni, L.: Unsupervised meta-learning through latent-space interpolation in generative models. *CoRR abs/2006.10236* (2020)
22. Krizhevsky, A.: Learning Multiple Layers of Features from Tiny Images, pp. 32–33 (2009)
23. Lake, B.M., Salakhutdinov, R., Tenenbaum, J.B.: Human-level concept learning through probabilistic program induction. *Science* **350**(6266), 1332–1338 (2015)
24. Lee, K., Maji, S., Ravichandran, A., Soatto, S.: Meta-learning with differentiable convex optimization. In: CVPR, pp. 10657–10665 (2019)
25. Li, J., Liu, G.: Few-shot image classification via contrastive self-supervised learning. *arXiv abs/2008.09942* (2020)
26. Li, J., Wang, Z., Hu, X.: Learning intact features by erasing-inpainting for few-shot classification. In: AAAI, pp. 8401–8409 (2021)
27. Li, S., Liu, F., Jiao, L., Chen, P., Li, L.: Self-supervised self-organizing clustering network: a novel unsupervised representation learning method. *IEEE Trans. Neural Netw. Learn. Syst.* pp. 1–15 (2022)
28. Li, W., Xu, J., Huo, J., Wang, L., Gao, Y., Luo, J.: Distribution consistency based covariance metric networks for few-shot learning. In: AAAI, pp. 8642–8649 (2019)
29. Li, Z., Zhou, F., Chen, F., Li, H.: Meta-sgd: learning to learn quickly for few shot learning. *CoRR abs/1707.09835* (2017)
30. Liu, C., Fu, Y., Xu, C., Yang, S., Li, J., Wang, C., Zhang, L.: Learning a few-shot embedding model with contrastive learning. In: AAAI, pp. 8635–8643 (2021)
31. Liu, F., Qian, X., Jiao, L., Zhang, X., Li, L., Cui, Y.: Contrastive learning-based dual dynamic GCN for SAR image scene classification. *IEEE Trans. Neural Netw. Learn. Syst.* pp. 1–15 (2022)
32. Lu, J., Gong, P., Ye, J., Zhang, C.: Learning from very few samples: a survey. *CoRR abs/2009.02653* (2020)
33. Medina, C., Devos, A., Grossglauser, M.: Self-supervised prototypical transfer learning for few-shot classification. *CoRR abs/2006.11325* (2020)
34. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) *ECCV 2016*. LNCS, vol. 9910, pp. 69–84. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46466-4_5

35. Paszke, A., et al.: Pytorch: an imperative style, high-performance deep learning library. In: NeurIPS, pp. 8024–8035 (2019)
36. Qin, T., Li, W., Shi, Y., Gao, Y.: Unsupervised few-shot learning via distribution shift-based augmentation. CoRR abs/2004.05805 (2020)
37. Ravi, S., Larochelle, H.: Optimization as a model for few-shot learning. In: ICLR (2017)
38. Ren, M., et al.: Meta-learning for semi-supervised few-shot classification. In: ICLR (2018)
39. Russakovsky, O., et al.: ImageNet large scale visual recognition challenge. *Int. J. Comput. Vis.* **115**(3), 211–252 (2015). <https://doi.org/10.1007/s11263-015-0816-y>
40. Shen, Z., Liu, Z., Qin, J., Savvides, M., Cheng, K.: Partial is better than all: revisiting fine-tuning strategy for few-shot learning. In: AAAI, pp. 9594–9602 (2021)
41. Snell, J., Swersky, K., Zemel, R.S.: Prototypical networks for few-shot learning. In: NIPS, pp. 4077–4087 (2017)
42. Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P.H.S., Hospedales, T.M.: Learning to compare: Relation network for few-shot learning. In: CVPR, pp. 1199–1208 (2018)
43. Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., Wierstra, D.: Matching networks for one shot learning. In: NIPS, pp. 3630–3638 (2016)
44. Wang, Y.-X., Hebert, M.: Learning to learn: model regression networks for easy small sample learning. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9910, pp. 616–634. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46466-4_37
45. Xie, J., Girshick, R.B., Farhadi, A.: Unsupervised deep embedding for clustering analysis. In: ICML, vol. 48, pp. 478–487 (2016)
46. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 649–666. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46487-9_40
47. Zhang, W., Jiao, L., Liu, F., Yang, S., Song, W., Liu, J.: Sparse feature clustering network for unsupervised SAR image change detection. *IEEE Trans. Geosci. Remote Sens.* **60**, 1–13 (2022)
48. Zhang, X., Jiao, L., Liu, F., Bo, L., Gong, M.: Spectral clustering ensemble applied to SAR image segmentation. *IEEE Trans. Geosci. Remote Sens.* **46**(7), 2126–2136 (2008)