UCCC: Unsupervised Community-consensus Contrastive Clustering

Anonymous Author(s)
Affiliation
Address
email

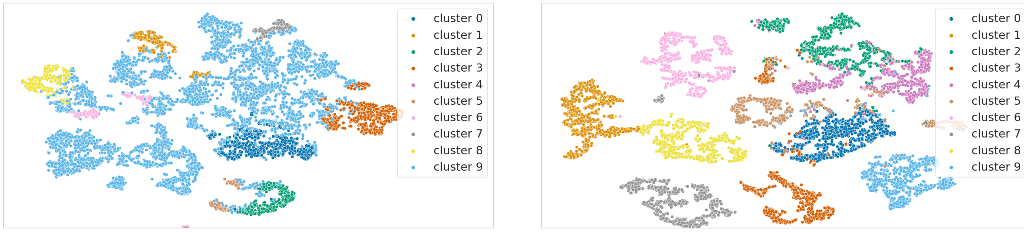
Abstract

As one of the most important tasks in computer vision, unsupervised image classification aims to group images into semantically meaningful clusters without using any labels. In this paper, we propose a one-stage clustering method called *Unsupervised Community-consensus Contrastive Clustering* (UCCC), which performs both instance- and cluster-level contrastive learning. In our framework, instance-level contrastive learning is capable of learning discriminative features, thereby helping construct reliable communities; cluster-level contrastive learning is conducted with the aid of the established communities, and further produces community-consensus cluster predictions. In particular, we design a novel instance-based loss function for the cluster-level contrastive learning. We demonstrate analytically that the gradient of our loss function could alleviate cluster degeneracy and thus prevent from a trivial solution, where the clusters are collapsed into a single entity. Extensive experimental results show that UCCC consistently outperforms state-of-the-art methods on six benchmark datasets.

1 Introduction

Deep neural networks have achieved human-level accuracy in image classification with the aid of large-scale datasets that contain annotated images, i.e. images with their corresponding semantic label. Nevertheless, annotating sufficient data is labor-intensive and time-consuming, establishing significant barriers for adapting the image classification systems to new domains. As a result, the focus of researchers is shifting to how to tackle image classification in an unsupervised manner. Some works [1–4] utilize the architecture of neural networks as a prior to cluster images, and refine the clusters iteratively by deriving the supervisory signal from the most confident sample [1, 2] or through cluster re-assignments calculated offline [3, 4]. Though this kind of two-stage methods could jointly learn representations and perform clustering, the errors accumulated during the alternation might result in sub-optimal clustering performance. On the other hand, the newly proposed CC [5] manages to learn discriminative representation and perform clustering simultaneously. The key idea of CC is to consider both instance- and cluster-level contrastive learning. To be specific, for a given a dataset, the positive and negative instance pairs are constructed through data augmentations, where the positive one is composed of two augmented views of the same instance and the other pairs are defined to be negative. By gathering the positive pairs and scattering the negatives pairs, the instance- and cluster-level contrastive learning are conducted in the row and column space of the feature matrix. In other words, the rows and columns of a feature matrix are regarded as the instance representations and cluster representations, respectively. However, using cluster representations for cluster-level contrastive learning may aggravate cluster degeneracy [3] and thus lead to a trivial solution, where the clusters are collapsed into a single entity. To deal with the aforementioned issue, CC requires an additional balance loss to maximize the entropy of cluster assignment probabilities, which however turns out to be the competing between the contrastive loss and the balance term. In contrast, we

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.



(a) CC (b) UCCC (ours)

Figure 1: The t-SNE visualization of cluster-based contrastive clustering like CC [5] and our proposed instance-based contrastive clustering UCCC. Both models are trained without the balance term. It can be observed CC (Fig. 1a) suffers from severe degeneracy problem while UCCC (Fig.1b), as an instanced-based contrastive clustering method, has no such issue.)

propose a one-stage clustering method called *Unsupervised Community-consensus Contrastive Clustering* (UCCC), which also adopts a dual contrastive learning framework but introduces a novel instance-based loss function for cluster-level contrastive learning.

Since objects in the same group are more similar to each other than to those in other groups, similar-looking objects usually belong to the same cluster while objects that can be easily distinguished tend to come from different clusters. Motivated by this observation, for a given instance, we expect its cluster prediction close to the one estimated based on its similar-looking positive community; on the other hand, the unlike-looking negative community should give to different cluster prediction. In our framework, instance-level contrastive learning is capable of learning discriminative features, thereby helping construct reliable communities; cluster-level contrastive learning is conducted with the aid of the established communities, and further produces community-consensus cluster prediction. In particular, we design the cluster-level contrastive loss function to: 1) maximize the prediction similarity between the positive instance pairs, and 2) minimize the positive-negative prediction similarity between the negative instance pairs. Doing so allows us to encourage not only self-consistent cluster prediction but also consensual cluster predictions of community.

The main contributions of this work are as follows: (1) We propose a one-stage clustering method called *Unsupervised Community-consensus Contrastive Clustering* (UCCC), in which a novel instance-based loss function for cluster-level contrastive learning is introduced. (2) We analytically compare the cluster- and instance-based contrastive loss function, showing the former to be a factor of data collapse while the latter could effectively alleviate this problem. Empirical results also prove that our approach does not suffer from data collapse due to the designed instance-based loss function. (3) Extensive experiments on six benchmark datasets show that our approach outperforms state-of-the-art methods in terms of three widely used clustering metrics, i.e., normalized mutual information (NMI), adjusted rand index (ARI) and accuracy (ACC).

2 Related Work

Unsupervised Image Clustering. As one of the most important tasks in computer vision, unsupervised image classification aims to group images into semantically meaningful clusters without using any labels. Recently, Van Gansbeke et al. propose a two-staged approach, SCAN, where feature learning and clustering are decoupled. In particular, SCAN first employs a self-supervised task [7] to obtain high-level feature representations then clusters those representations by nearest neighbors. Another work that incorporates self-supervised representation learning into clustering is CC [5], which takes semantic labels as a special representation and conducts the instance- and the cluster-level contrastive learning simultaneously. In this work we adopt a dual contrastive framework like CC as multi-staged approaches take much more time to deploy and hardly achieves improvements compared with one-stage methods. The main difference between CC and ours is in how we perform cluster-level contrastive learning. CC conducts cluster-level learning in the column space of representation vectors, which would aggravate cluster degeneracy and thus require additional balance term. Instead, we perform cluster-level contrastive learning in the row space of representations by adopting a novel loss function, leading to substantial improvements in the clustering performance.

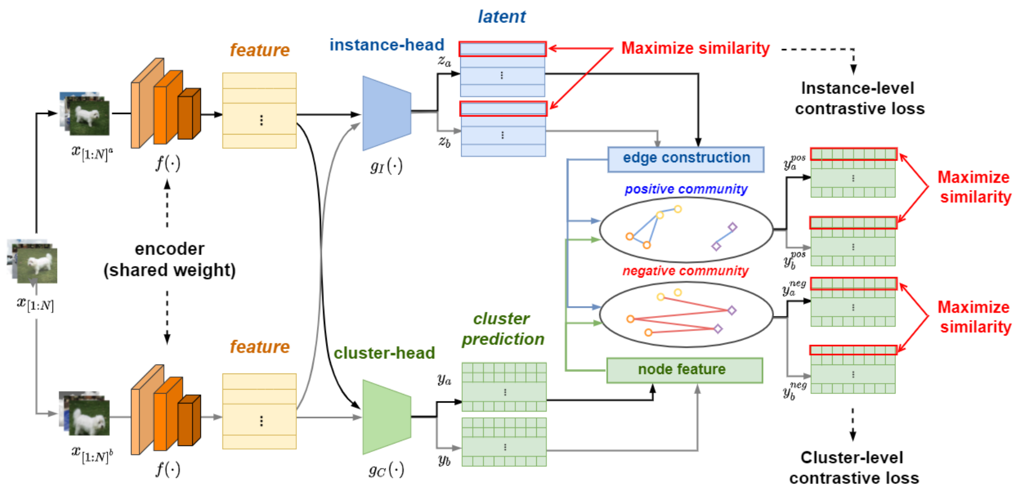


Figure 2: Our unsupervised community-consensus contrastive clustering (UCCC) framework consists of an encoder f and two MLPs that correspond to an instance head g_I and a cluster head g_C. Two random data augmentations are applied on each input image to obtain data pairs. Given data pairs, the shared encoder is used to extract features from different augmentations. These feature are projected into two subspace to conduct instance- and cluster-level contrastive learning using the corresponding projection head. Taking the features in two subspace as prior, two community graph are constructed to produce consensual cluster predictions with the guidance of the cluster-level contrastive loss.

3 Unsupervised Community-Consensus Contrastive Clustering

As shown in Fig. 2 our model consist of two sub-branch: the Instance-level Contrastive Branch (ICB) and the Cluster-level Contrastive Branch (CCB). Both branches take the output of the encoder $f(\cdot)$ as input. ICB projects the input feature into a low-dimensional space where the corresponding contrastive loss is applied. The discriminative features learned by ICB can not only attain inter-cluster difference but also preserve intra-cluster distinction, and thus help construct reliable instance communities for CCB. Since the semantic label can be regarded as a special representation, CCB projects the input instances into a subspace with a dimensionality of the cluster number, and consensual cluster predictions are further achieved with the guidance of our proposed contrastive loss. In the following, we will describe how we perform a dual contrastive learning for unsupervised clustering in detail and introduce the proposed objective function at the end.

3.1 Instance-level Contrastive Learning

To learn representations without labels, we leverage a self-supervised approach SimCLR [7], which uses "instance discrimination" as a pretext task. Given a minibatch of images $\{\boldsymbol{x}_i\}_{i=1}^{N}$, we apply random image transformations (e.g., cropping or blurring) twice on each image, thus generating two different view of them (augmentation a and b). The transformed images are projected to a subspace via $\boldsymbol{z} = g_I(f(\boldsymbol{x}))$, where g_I is a two-layer MLP projection head. The resulting 2N data points $\{\boldsymbol{z}_{1^a}, \boldsymbol{z}_{2^a}, ..., \boldsymbol{z}_{N^a}, \boldsymbol{z}_{1^b}, \boldsymbol{z}_{2^b}, ..., \boldsymbol{z}_{N^b}\}$ will be used to calculate the contrastive loss as described below.

Instance-level Contrastive Loss. The common idea of contrastive learning is the following: pull together an anchor and a "positive" sample (minimize the similarity between a positive pair), and push apart the anchor from many "negative" samples (maximize the similarity between negative pairs). A positive pair often consists of instances augmented from the same sample, and negative pairs are formed by the anchor and randomly chosen samples from the minibatch. Here, we use the dot product between $L2$ normalized features, which is cosine similarity, as the metric for instance similarity. Let $\tilde{\boldsymbol{z}} = \frac{\boldsymbol{z}}{\|\boldsymbol{z}\|}$ denote the normalized feature. Then instance-level contrastive loss for an anchor i is defined as:

$$\ell_{i^a} = -\log \frac{exp(\tilde{\boldsymbol{z}}_{i^a} \cdot \tilde{\boldsymbol{z}}_{i^b}/\tau)}{\Sigma_{j=1}^{N} \mathbb{1}_{[j \neq i]} [\, exp(\tilde{\boldsymbol{z}}_{i^a} \cdot \tilde{\boldsymbol{z}}_{j^a}/\tau) + exp(\tilde{\boldsymbol{z}}_{i^a} \cdot \tilde{\boldsymbol{z}}_{j^b}/\tau)\,]}, \tag{1}$$

where $\mathbb{1}_{[j \neq i]} \in \{0, 1\}$ is an indicator function evaluating to 1 iff $j = i$ and τ is the instance-level temperature parameter. Considering every augmented samples within a minibatch, the instance-level loss is computed as:

$$\mathcal{L}_{ins} = \sum_{i=1}^{N} \ell_{i^a} + \ell_{i^b}. \tag{2}$$

3.2 Cluster-level Contrastive Learning

In cluster-level learning, we leverage the concept of "community consensus". That is, for a given instance, its similar-looking positive peers would come from the same cluster as itself; on the other hand, the unlike-looking negative peers should belong to different clusters. Following this idea, we first construct a positive graph and a negative graph to present the community relations.

Community Peers. Image distinction is effectively preserved in the feature space learned by ICB since the instance-level contrastive loss encourages high similarity only between the instances augmented from the same image. Motivated by this observation, we use the similarities of representations learned by the instance head to construct positive/negative communities.

Table 1: Positive confidence threshold with respect to the number of clusters.

# of clusters	10	15	20	200
pos_conf	0.258	0.484	0.615	0.961

Specifically, we regard the community relations as a graph structure and define the positive and negative adjacency matrices, $\boldsymbol{A}_{pos}, \boldsymbol{A}_{neg} \in \mathbb{R}^{2N \times 2N}$. As there might be ambiguous pairs that are not similar enough, a confidence threshold is required to filter out noisy information. We set the positive confidence threshold by the cosine value of an angle θ, where θ is calculated based on the number of clusters, C. Larger C usually implies more hardly distinguishable noisy pairs exist, and thus requires higher positive confidence. Table 1 summarizes the considered positive confidence in this work. See supplementary Sec. A for more details about the angle θ. The elements of positive adjacency matrix are thereby defined as:

$$A_{ij,pos} = \begin{cases} sim(\tilde{\boldsymbol{z}}_i, \tilde{\boldsymbol{z}}_j), & if\ sim(\tilde{\boldsymbol{z}}_i, \tilde{\boldsymbol{z}}_j) > pos\_conf \\ 0, & otherwise \end{cases}, \tag{3}$$

where the augmentation notations a, b of features $\tilde{\boldsymbol{z}}_i, \tilde{\boldsymbol{z}}_j$ are ignored for simplification. Similarly, the elements of negative adjacency matrix are calculated as shown in Eq. 4. Since diverse negative peers could help the model learn how to distinguish the belonging cluster from other clusters, the negative confidence is set as 0, which is a relatively low value in comparison with the positive one.

$$A_{ij,neg} = \begin{cases} -sim(\tilde{\boldsymbol{z}}_i, \tilde{\boldsymbol{z}}_j), & if\ sim(\tilde{\boldsymbol{z}}_i, \tilde{\boldsymbol{z}}_j) < neg\_conf \\ 0, & otherwise \end{cases} \tag{4}$$

Assignment Aggregation. To produce cluster assignments for given samples, the input features are mapped into a subspace with a dimensionality of the cluster number via $\boldsymbol{y} = g_C(f(\boldsymbol{x}))$, where $\boldsymbol{y} \in \mathbb{R}^C$ denotes a cluster assignment (i-th element can be interpreted as the probability of sample $\boldsymbol{x}$ being assigned to the cluster i) and g_c is a two-layer projection head. Now that the community graphs are established, we would like aggregate the information from community relations, and estimate a positive and a negative cluster assignment for each instance according to its corresponding peers. For a mini-batch of augmented samples, its cluster assignments $\{\boldsymbol{y}_{i^m}\}_{i=1, m\in\{a,b\}}^{N}$ can be concatenated into a matrix $\boldsymbol{Y} \in \mathbb{R}^{2N \times C}$. We consider k-hop peer relations and compute the positive/negative assignment matrix inspired from [8]:

$$\begin{cases} \boldsymbol{Y}_{pos} = \left(\prod_{i=1}^{k} \tilde{\boldsymbol{A}}_{pos}\right) \boldsymbol{Y} \\ \\ \boldsymbol{Y}_{neg} = \tilde{\boldsymbol{A}}_{neg} \left(\prod_{i=1}^{k-1} \tilde{\boldsymbol{A}}_{pos}\right) \boldsymbol{Y} \end{cases}, \tag{5}$$

Algorithm 1: Unsupervised Community-consensus Contrastive Clustering

Input: Dataset $\mathcal{X}$, cluster number C, Training steps S, Batch size N, Temperature τ, confidence threshold δ, structure of encoder f, instance head g_I, and cluster head g_C

Output: cluster assignments of dataset $\mathcal{X}$

/* training */

for *step s = 1 to S* **do**

- sample a minibatch $\{\boldsymbol{x}_i\}_{i=1}^N$ from $\mathcal{X}$
- obtain $2N$ augmented samples $\{\boldsymbol{x}_{i^m}\}_{i=1,m\in\{a,b\}}^N$ through data augmentation
- compute instance representations and cluster assignments by $\boldsymbol{z}_{i^m} = g_I(f(\boldsymbol{x}_{i^m}))$ and $\boldsymbol{y}_{i^m} = g_C(f(\boldsymbol{x}_{i^m}))$
- construct the positive/negative adjacency matrix through Eq. 3-4
- compute positive and negative cluster assignments through Eq. // community fusion
- compute the instance-level loss $\mathcal{L}_{ins}$ and cluster-level loss $\mathcal{L}_{clu}$ through Eq. 1-2, 6
- compute overall loss $\mathcal{L}$ by Eq. 7
- update f, g_I and g_C to minimize $\mathcal{L}$

/* testing */

for $\boldsymbol{x} \in \mathcal{X}$ **do**

- assign $\boldsymbol{x}$ to cluster $c = \arg\max g_C(f(\boldsymbol{x}))$

with the normalized adjacency matrices $\tilde{\boldsymbol{A}}_* = \boldsymbol{D}_*^{-\frac{1}{2}} \boldsymbol{A}_* \boldsymbol{D}_*^{-\frac{1}{2}}, * \in \{pos, neg\}$ and $\boldsymbol{D}$ denoting a degree matrix where $D_{ii} = \sum_j A_{ij}$.

Community-consensus Contrastive Clustering Loss. For an instance, the positive cluster assignment can be viewed as its cluster assignment while further taking positive peers into consideration; the negative cluster assignment, on the other hand, corresponds to the least possible cluster assignment that it would have. In this sense, we follow the idea of "community consensus" and propose a novel instance-based contrastive loss as shown in Eq. 6. In particular, the proposed loss function encourages self-consistent predictions by maximize the similarity between the assignments of positive instance pairs (augmented from the same image). Moreover, consensual predictions are achieved by minimizing the similarity between the positive and the negative assignments of all instance pairs (any two of instances within a mini-batch).

$$\mathcal{L}_{clu} = -\log \frac{\sum_{i=1}^{N} \exp\left(\boldsymbol{y}_{i^a,pos} \cdot \boldsymbol{y}_{i^b,pos} + \boldsymbol{y}_{i^a,neg} \cdot \boldsymbol{y}_{i^b,neg}\right)}{\sum_{\forall (i,j,m,n)\in\Phi} \exp\left(\boldsymbol{y}_{i^m,pos} \cdot \boldsymbol{y}_{j^n,neg}\right)} , \tag{6}$$

where $\Phi = \{(i,j,m,n) \mid \forall\, i,j \in \{1,..,N\}\ and\ \forall\, m,n \in \{a,b\}\}$.

3.3 Overall Objective

Algorithm 1 summarizes the full training and test process of the model. Both ICB and CCB are simultaneously optimized by the corresponding loss in an end-to-end manner. Hence, the overall objective function is written as:

$$\mathcal{L} = \mathcal{L}_{ins} + \mathcal{L}_{clu} \tag{7}$$

4 Contrastive Losses for clustering: cluster-based and instance-based

In this section, we look at the cluster-based contrastive loss proposed in [5] and our instance-based clustering loss, showing why our formulation is superior to the former one.

Formally, let $\boldsymbol{Y}_m \in \mathbb{R}^{N\times C}$ be the cluster assignments for a mini-batch under some augmentation m. Since each sample belongs to only one cluster, the rows of $\boldsymbol{Y}_m$ tends to be one-hot. In this sense, the i-th column of $\boldsymbol{Y}_m$, denoted $\boldsymbol{y}_m^i$, can be viewed as a representation of the i-th cluster. For each cluster c, a positive cluster pair is formed by combining $\boldsymbol{y}_a^c$ and $\boldsymbol{y}_b^c$, namely, the representations under two different augmentation, while other $2C-2$ pairs are considered to be negative. To minimize the inter-cluster similarities to separate different clusters, the cluster-based contrastive loss is written as:

$$\mathcal{L}_{clu}^{clu} = -\sum_{\substack{c=1,\\ m\in\{a,b\}}}^{C} \log \frac{\exp\left(\boldsymbol{y}_a^c \cdot \boldsymbol{y}_b^c\right)}{\sum_{j=1}^{C} \mathbb{1}_{[i\neq c]} \left[\exp\left(\boldsymbol{y}_m^c \cdot \boldsymbol{y}_a^j\right) + \exp\left(\boldsymbol{y}_m^c \cdot \boldsymbol{y}_b^j\right)\right]} \tag{8}$$

It is observed that the cluster-based loss is conducted in the column space of assignment matrices, which is different from our instance-based clustering loss (Eq. 6) that measures the similarity in the row space.

As stated in [5] the cluster-based contrastive loss requires the following balance term to avoid the trivial solution that most instances are assigned to the same cluster.

$$\mathcal{L}_{balance} = \sum_{c=1}^{C} \left[P(\boldsymbol{y}_a^c)\log P(\boldsymbol{y}_a^c) + P(\boldsymbol{y}_b^c)\log P(\boldsymbol{y}_b^c)\right] \tag{9}$$

Here, $\mathcal{L}_{balance}$ is the negative entropy of assignment probabilities $P(\boldsymbol{y}_m^c) = \sum_{i=1}^{N} y_{im}^c / \|\boldsymbol{Y}\|_1, m \in \{a, b\}$ within a mini-batch under each data augmentation. Such kind of entropy maximization would lead to a sub-optimal clustering result since there is no guarantee that all clusters within a dataset should be equal-sized. Our instance-based loss in contrast with $\mathcal{L}_{clu}^{clu}$ does not suffer from this issue and is capable of alleviating degeneracy. An analysis of the gradients with respect to the weights of the last layer of the cluster head $\boldsymbol{w}^c$ (can be interpreted as a classifier for cluster c) supports this conclusion. Throughout the analysis, we assume the following assumption for simplification:

Assumption 1 (Generalization)**.** Let a, b be any two random augmentations. The statements below hold:

$$\begin{cases} \boldsymbol{y}_a^j \approx \boldsymbol{y}_b^j \approx \hat{\boldsymbol{y}}^j & \text{for} \quad j \in \{1, ..., C\} \\ \boldsymbol{H}_a \approx \boldsymbol{H}_b \approx \hat{\boldsymbol{H}} \end{cases},$$

where $\boldsymbol{H}_m$ denotes the embedding matrix (prior to the last layer of cluster head) for a mini-batch under augmentation m.

Let $\boldsymbol{Y}_{m,pos}$ be the positive cluster assignments for a mini-batch under augmentation m and the corresponding embedding matrix $\boldsymbol{H}_{m,pos}$ is a combination of transformed $\boldsymbol{H}_a$ and $\boldsymbol{H}_b$ as positive cluster assignments are simply linear combinations of original cluster assignments within a mini-batch. In a similar fashion, the negative cluster assignments for a mini-batch under augmentation m and its embedding matrix are denoted by $\boldsymbol{Y}_{m,neg}$ and $\boldsymbol{H}_{m,neg}$ respectively. It is noted that the assumption of model generalization also implies:

$$\begin{cases} \boldsymbol{y}_{a,*}^j \approx \boldsymbol{y}_{b,*}^j \approx \hat{\boldsymbol{y}}_*^j & \text{for} \quad j \in \{1, ..., C\},\ * \in \{pos, neg\} \\ \boldsymbol{H}_{a,*} \approx \boldsymbol{H}_{b,*} \approx \hat{\boldsymbol{H}}_* \end{cases}$$

As shown in the Supplementary, the gradient for our clustering loss $\mathcal{L}_{clu}$ is derived as:

$$\frac{\partial \mathcal{L}_{clu}}{\partial \boldsymbol{w}^c} \approx -\frac{2}{N}\left[\hat{\boldsymbol{H}}_{pos}^{\intercal}\hat{\boldsymbol{y}}_{pos}^c + \hat{\boldsymbol{H}}_{neg}^{\intercal}\hat{\boldsymbol{y}}_{neg}^c\right] + \frac{1}{\Psi}\left[\left(\hat{\boldsymbol{E}}\hat{\boldsymbol{H}}_{pos}\right)^{\intercal}\hat{\boldsymbol{y}}_{neg}^c + \left(\hat{\boldsymbol{E}}\hat{\boldsymbol{H}}_{neg}\right)^{\intercal}\hat{\boldsymbol{y}}_{pos}^c\right], \tag{10}$$

with $\Psi = \sum_{i=1}^{N}\sum_{j=1}^{N} \exp\left(\hat{\boldsymbol{y}}_{i,pos} \cdot \hat{\boldsymbol{y}}_{j,neg}\right)$ and $\hat{\boldsymbol{E}} \in \mathbb{R}^{N\times N}$ where $\hat{E}_{ij} = \exp\left(\hat{\boldsymbol{y}}_{i,pos} \cdot \hat{\boldsymbol{y}}_{j,neg}\right)$.

The discussion about the cluster-based contrastive loss is in the Supplementary.

Discussion. Since we use gradient descent [9] as our optimization algorithm, the classifiers $\boldsymbol{w}^c$ is updated through $\boldsymbol{w}^c = \boldsymbol{w}^c - \eta\,\frac{\partial \mathcal{L}_{clu}}{\partial \boldsymbol{w}^c}$ every step. To understand why our formulation does not lead to cluster degeneracy, we make the following observations. First, if an instance has a high assignment score (either positive or negative) with respect to a cluster c, the negative term in Eq. 10 would pull the classifier $\boldsymbol{w}^c$ close to its corresponding embedding features while the positive term would push $\boldsymbol{w}^c$ away from its opposite embedding. Second, because the community graph is established based on instance-level features, the community peers are sufficiently reliable to estimate positive/negative assignment, which guarantees $\hat{\boldsymbol{Y}}_{neg}$ is not a zero matrix. This makes sure the growth of each cluster and prevents most instances from falling into the same entity.

Table 2: State-of-the-art comparison: The performance (%) of our model are reported in **bold** font. For fair comparison with SCAN [6], we also report the performance of our method with ResNet18 as the backbone of encoder.

Dataset	CIFAR-10			CIFAR-20			STL-10			ImageNet-10			ImageNet-Dogs			tiny-ImageNet		
Metrics	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC
K-means [15]	8.7	4.9	22.9	8.4	2.8	13.0	12.5	6.1	19.2	11.9	5.7	24.1	5.5	2.0	10.5	6.5	0.5	2.5
SC [16]	10.3	8.5	24.7	9.0	2.2	13.6	9.8	4.8	15.9	15.1	7.6	27.4	3.8	1.3	11.1	6.3	0.4	2.2
AC [17]	10.5	6.5	22.8	9.8	3.4	13.8	23.9	14.0	33.2	13.8	6.7	24.2	3.7	2.1	13.9	6.9	0.5	2.7
NMF [18]	8.1	3.4	19.0	7.9	2.6	11.8	9.6	4.6	18.0	13.2	6.5	23.0	4.4	1.6	11.8	7.2	0.5	2.9
AE [19]	23.9	16.9	31.4	10.0	4.8	16.5	25.0	16.1	30.3	21.0	15.2	31.7	10.4	7.3	18.5	13.1	0.7	4.1
DAE [20]	25.1	16.3	29.7	11.1	4.6	15.1	22.4	15.2	30.2	20.6	13.8	30.4	10.4	7.3	18.5	12.7	0.7	3.9
DCGAN [21]	26.5	17.6	31.5	12.0	4.5	15.1	22.4	15.2	30.2	22.5	15.7	34.6	12.1	7.8	17.4	13.5	0.7	4.1
DeCNN [22]	24.0	17.4	28.2	9.2	3.8	13.3	22.7	16.2	29.9	18.6	14.2	31.3	9.8	7.3	17.5	11.1	0.6	3.5
VAE [23]	24.5	16.7	29.1	10.8	4.0	15.2	20.0	14.6	28.2	19.3	16.8	33.4	10.7	7.9	17.9	11.3	0.6	3.6
JULE [24]	19.2	13.8	27.2	10.3	3.3	13.7	18.2	16.4	27.7	17.5	13.8	30.0	5.4	2.8	13.8	10.2	0.6	3.3
DEC [1]	25.7	16.1	30.1	13.6	5.0	18.5	27.6	18.6	35.9	28.2	20.3	38.1	12.2	7.9	19.5	11.5	0.7	3.7
DAC [2]	39.6	30.6	52.2	18.5	8.8	23.8	36.6	25.7	47.0	39.4	30.2	52.7	21.9	11.1	27.5	19.0	1.7	6.6
ADC [25]	-	-	32.5	-	-	16.0	-	-	53.0	-	-	-	-	-	-	-	-	-
DDC [26]	42.4	32.9	52.4	-	-	-	37.1	26.7	48.9	43.3	34.5	57.7	-	-	-	-	-	
DCCM [27]	49.6	40.8	62.3	28.5	17.3	32.7	37.6	26.2	48.2	60.8	55.5	71.0	32.1	18.2	38.3	22.4	3.8	10.8
IIC * [28]	-	-	61.7	-	-	25.7	-	-	59.6	-	-	-	-	-	-	-	-	-
EmbedUL [29]	-	-	81.0	-	-	35.3	-	-	66.5	-	-	-	-	-	-	-	-	-
PICA [30]	59.1	51.2	69.6	31.0	17.1	33.7	61.1	53.1	71.3	80.2	76.1	87.0	35.2	20.1	35.2	27.7	4.0	9.8
CC * [5]	70.5	63.7	79.0	43.1	26.6	42.9	76.4	72.6	85.0	85.9	82.2	89.3	44.5	27.4	42.9	34.0	7.1	14.0
SCAN [6]	71.5	66.5	81.6	44.9	28.3	44.0	67.3	61.8	79.2	-	-	-	-	-	-	-	-	-
UCCC (Ours) * ⇓																		
Res18 *	**74.8**	**71.0**	**84.2**	**47.4**	**31.3**	**46.9**	**73.7**	**70.8**	**84.7**	**87.9**	**88.4**	**94.5**	**48.9**	**36.5**	**52.2**	**36.8**	**10.4**	**20.4**
Res34 *	**76.8**	**73.2**	**85.5**	**49.3**	**33.1**	**48.4**	**76.8**	**74.3**	**86.7**	**89.3**	**89.9**	**95.3**	**50.9**	**38.5**	**53.5**	**39.1**	**11.8**	**22.4**

*: one-staged clustering method.

5 EXPERIMENTS

The experimental evaluation is performed on six benchmark datasets, i.e. CIFAR-10, CIFAR-100 [10], STL-10 [11], ImageNet-10, ImageNet-Dogs [2], and tiny-ImageNet [12]. The first two CIFAR datasets contain 60,000 images of 32x32 pixels. Following previous works [6, 5], we take 20 super-classes rather than 100 classes as the ground-truth for CIFAR-100. The next is STL-10 containing 13,000 labeled images and 100,000 unlabeled images of 96x96 pixels. The additional 100,000 unlabeled images are used to perform the instance-level contrastive learning. For the last three ImageNet datasets, only the training set is used. ImageNet-10 contains 13,000 images of 10 classes, ImageNet-Dogs consists of 19,500 images from 15 dog classes, and large-scaled tiny-ImageNet contains 100,000 images from 200 classes.

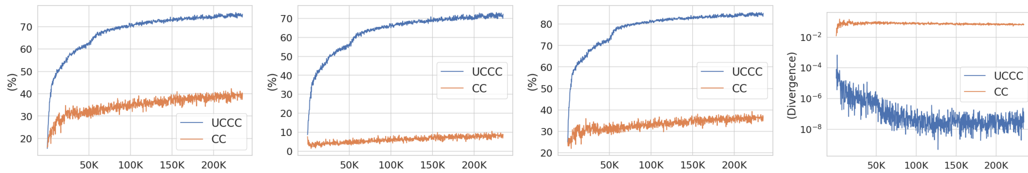
We adopt ResNet18/ResNet34 [13] as our backbone net and modify the stride of the first convolution layer to 1, which enables the encoder to extract more delicate features. Especially for small-sized datasets (i.e. CIFAR), we remove the first maxpooling layer and replace the activation of the first layer to Mish [14]. For ICB, the dimensionality of the final embedding space is set to 128, and the temperature parameter τ is fixed to 0.5 in all experiments. For CCB, the dimensionality of cluster assignments is naturally set to the number of clusters, and the assignment aggregation is fixed as 2-hop graph fusion in all used datasets. We set the batch size to $N = 256 \times n\_class/10$ except for tiny-ImageNet, where N is set to 512 due to the memory limitation, and the image size is set to 32 for CIFAR datasets, 224 for other datasets. The whole model is trained from scratch for 1,000 epochs on NVIDIA Tesla V100 GPU.

Evaluation criterion. We evaluate the results based on three widely used clustering metrics including normalized mutual information (NMI), adjusted rand index (ARI) and accuracy (ACC). To further analyze the severity of cluster degeneracy, we also report Jensen–Shannon divergence (JSD) in Sec 5.2. Except for JSD, higher values of these metrics indicate better clustering performance.

5.1 Comparison with state-of-the-arts

We compare our method to the state-of-the-art on six different benchmarks. The compared state-of-the-art are mostly multi-staged methods, e.g. SC [16], NMF [18], AE [19], DAE [20], DCGAN [21], DeCNN [22], and VAE [23] obtain clustering results via k-means on the features extracted from images. According to the results shown in Table 2, UCCC outperforms these competing baselines by a large margin on all six datasets. Notably, UCCC obtains classification accuracy improvements compared with the closest competitor CC [5] by 6.5% on CIFAR-10, 5.5% on CIFAR-20, 1.7%

Figure 3: The performance evolution of UCCC and CC (w/o the balance loss) on CIFAR-10.



(a) NMI (b) ARI (c) ACC (d) JSD

on STL-10, 6.4% on ImageNet-10, 10.6% on ImageNet-Dogs. As CC adopts a dual contrastive framework like ours, the remarkable results prove the efficacy of our design for cluster-level learning.

5.2 Cluster Degeneracy

Many previous works [15, 5, 6] would lead to degenerate solutions because such solutions are saddle points of the adopted objective function (Fig. 4). To avoid cluster degeneracy, they usually manage to assign an equal number of samples to each cluster by maximizing the entropy of clustering result. However, this kind of optimization may not be a general solution especially when it deals with imbalance clusters.

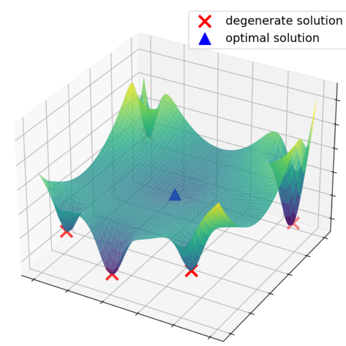


Figure 4: optimization surface

In Fig. 3, the performance curves of our approach and CC (w/o Eq. 9) during the training phase are presented to compare ours with the degenerate one. As the training process goes, the performances of UCCC in the four considered metrics are all improved, implying cluster assignments become more reasonable. By contrast, CC hardly achieves small improvements due to the degeneracy problem, which can be reflected by the high divergence between the distribution of the ground truth and its clustering. We further perform t-SNE in the instance-level space on both UCCC and CC (Fig. 1). The result well confirms our model does not suffer from cluster degeneracy since there is no dominant cluster in our result (Fig. 1b).

5.3 Ablation Study

5.3.1 Inductive Clustering

To further validate the effectiveness of UCCC, our method is evaluated on the three datasets under a more realistic inductive setting, where testing images are not available during the training phase. Specifically, we trained the model on the training set while measuring the performance using the images in testing split. Table 3 shows our relative performance drops over transductive setting. As can be seen, the performance drops are stable even though the used CNN backbones are different. In particular, the ACC drops on all three datasets are less than 3%, which well demonstrates the robustness of UCCC.

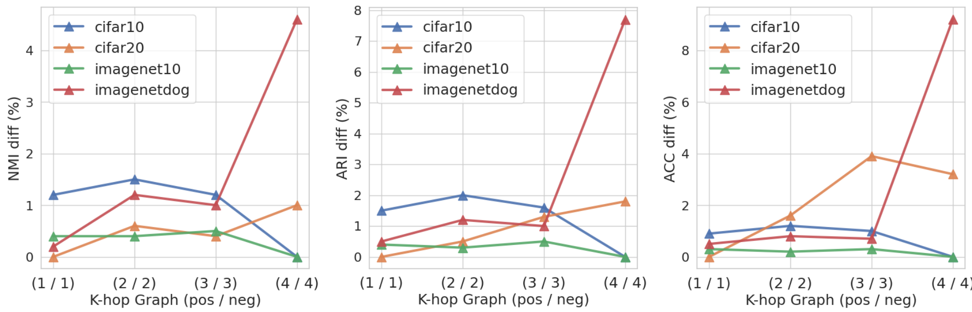
Table 3: The relative performance drops (%) under inductive setting over transductive setting.

Metrics	NMI	ARI	ACC
STL-10 (Res18)	-2.8	-3.7	-2.3
STL-10 (Res34)	-2.9	-3.4	-3.0
CIFAR-10 (Res18)	-2.4	-3.3	-2.0
CIFAR-10 (Res34)	-3.4	-2.8	-1.5
CIFAR-20 (Res18)	-3.8	-2.8	-1.1
CIFAR-20 (Res34)	-4.5	-3.8	-2.1

5.3.2 K-hop Graph Fusion

To study the effect of k-hop graph fusion, we take the performance with $k = 1$ as the baseline and present the trends of NMI/ARI/ACC gain with respect to varying values of k over {1,2,3,4} on four benchmark datasets. As shown in Fig. 5, we can see that performance improvements are achieved when considering more than 1-hop neighbors on the four datasets. It is also observed that using 2-hop graph fusion leads to the best performance in most cases because taking too many community peers into consideration usually comes with noisy information. Nevertheless, for ImageNet-Dogs dataset,

Figure 5: The influence of k-hop graph fusion on four datasets.



(a) NMI (b) ARI (c) ACC

the results using 4-hop graph fusion outperform the one with $k = 1$ by a large margin ($+3.4\%$ in NMI, $+6.5\%$ in ARI, $+8.4\%$ in ACC). The reason might be that ImageNet-Dogs consists of relatively similar images compared to other datasets, and thus more available information is beneficial to the model learning.

6 Conclusion

Based on the observation that similar-looking objects usually belong to the same cluster while objects that can be easily distinguished tend to come from different clusters, we propose the Unsupervised Community-consensus Contrastive Clustering (UCCC) which conducts the instance- and cluster-level contrastive learning simultaneously under a unified framework. By incorporating a novel instance-based contrastive loss into cluster-level learning, our model is encouraged to produce consensual cluster assignments. We further verify that our network does not lead to degenerate solutions due to the designed cluster-level contrastive loss. Experimental evaluation shows that the proposed method outperforms prior work by large margins for a variety of datasets.

Broader Impact

For unsupervised classification, there is no need to specify in advance all the classes in the image, which reduces the dependence of deep learning on massive labeled data. The proposed method in this work adopts contrastive learning to produce reliable cluster assignments. As a one-stage deep clustering method, our work is capable of extracting discriminative features and performing clustering through one-step training. Although such training process could reduce the deployment difficulty of clustering algorithms in practical applications, providing the opportunity to learn from big unannotated datasets may cause problems concerning users' information security and personal privacy if not controlled.

Broadly speaking, there are two shortcuts of deep clustering. Firstly, the prediction accuracy is still lower than models trained in a supervised manner. Thus clustering is not applicable to those settings that require high accuracy and confidence, e.g., self-driving cars. Secondly, computational complexity for deep clustering is much higher than the traditional algorithms, e.g. k-means. Thus the deep model can not be applied to applications where computational resource is limited.

References

[1] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487. PMLR, 2016.

[2] Jianlong Chang, Lingfeng Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Deep adaptive image clustering. In *Proceedings of the IEEE international conference on computer vision*, pages 5879–5887, 2017.

[3] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018.

[4] Mathilde Caron, Piotr Bojanowski, Julien Mairal, and Armand Joulin. Unsupervised pre-training of image features on non-curated data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2959–2968, 2019.

[5] Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. Contrastive clustering. *arXiv preprint arXiv:2009.09687*, 2020.

[6] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Scan: Learning to classify images without labels. In *European Conference on Computer Vision*, pages 268–285. Springer, 2020.

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[8] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[9] Shun-ichi Amari. Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5 (4-5):185–196, 1993.

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

[11] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

[12] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7:7, 2015.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[14] Diganta Misra. Mish: A self regularized non-monotonic activation function. *arXiv preprint arXiv:1908.08681*, 2019.

[15] 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.

[16] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering, 2004. *Advances in Neural Information Processing Systems*, 17, 2005.

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

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

[19] Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007.

[20] 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.

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

[22] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. Deconvolutional networks. In *2010 IEEE Computer Society Conference on computer vision and pattern recognition*, pages 2528–2535. IEEE, 2010.

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

[24] Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5147–5156, 2016.

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

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

[27] Jianlong Wu, Keyu Long, Fei Wang, Chen Qian, Cheng Li, Zhouchen Lin, and Hongbin Zha. Deep comprehensive correlation mining for image clustering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8150–8159, 2019.

[28] Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9865–9874, 2019.

[29] Sungwon Han, Sungwon Park, Sungkyu Park, Sundong Kim, and Meeyoung Cha. Mitigating embedding and class assignment mismatch in unsupervised image classification. In *16th European Conference on Computer Vision, ECCV 2020*. Springer, 2020.

[30] Jiabo Huang, Shaogang Gong, and Xiatian Zhu. Deep semantic clustering by partition confidence maximisation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8849–8858, 2020.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] see Sec. 4 and the Supplementary
   (b) Did you include complete proofs of all theoretical results? [Yes] see the Supplementary
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] see the Supplementary
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] see Sec. 5
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] see Sec. 5
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] see Sec. 5
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] reference code is in the Supplementary
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Positive Confidence Threshold

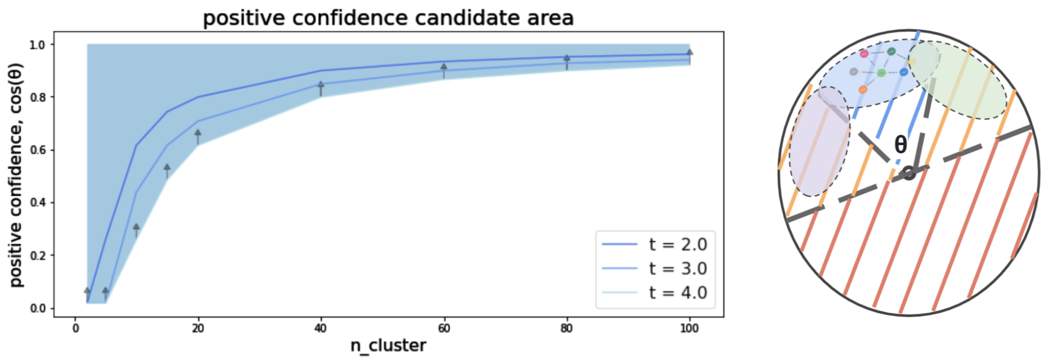


Figure 1: The left shows the positive confidence curve, $\cos\theta$, using different values of tolerant parameter t, and the blue area denotes the similarities that would be taken as positive when $t = 4.0$. The right illustrates the threshold θ, which is calculated based on the number of clusters and tolerant parameter t. The larger t means the higher tolerance for the overlapping (ambiguous) area between two different clusters.

As we take the cosine similarity (dot product between $L2$ normalized vectors) as our similarity metric, we use a threshold angle θ to filter out noisy information from ambiguous instance pairs. That is, if the similarity between an instance pair is lower than $\cos\theta$, we would regard it as a ambiguous pair.

To cluster data samples into groups, the feature vectors extracted from samples should be separable enough on the embedding space. Applying $L2$ normalization is actually projecting a feature vector into a point on unit sphere. For simplification, we consider the unit 2-sphere as the embedding sphere. Ideally, the embedding features are uniformly scattered on the surface of the sphere. We assume each cluster takes over the equal size of surface area and measure the surface area by taking each cluster as a spherical cap. Then for any given instance, we consider a spherical cap where the pole is its embedding point on the sphere, and regard the points on the cap as its positive peers. In other words, the threshold angle θ is calculated as:

$$\begin{aligned} 4\pi r^2 \cdot t &= n\_cluster \cdot 2\pi r^2 \left(1 - \cos\theta\right) \\ \Rightarrow \quad \cos\theta &= 1 - \tfrac{2t}{n\_cluster} \end{aligned} \tag{1}$$

Here t is a *tolerant parameter* controlling the tolerance for the intra-cluster overlapping area (Fig. 1 right). The tolerant parameter is set to 4 in all experiments as we empirically find it leads to the best performance in overall. See Sec. C.2 for the ablation study on the impact of t.

B Gradient Derivation

In this section, we present our derivation for the gradients of the two considered cluster-level contrastive losses, $\mathcal{L}_{clu}^{clu}$ and $\mathcal{L}_{clu}$, with respect to a classifier for cluster c, $\boldsymbol{w}^c$. The notations for derivations are summarized in Table 1. For convenience, we reprint below the expressions for each.

$$\mathcal{L}_{clu}^{clu} = - \sum_{\substack{c=1, \\ k\in\{a,b\}}}^{C} \log \frac{\exp\left(\boldsymbol{y}_a^c \cdot \boldsymbol{y}_b^c\right)}{\sum\limits_{j=1}^{C} \mathbb{1}_{[i\neq c]} \left[\exp\left(\boldsymbol{y}_k^c \cdot \boldsymbol{y}_a^j\right) + \exp\left(\boldsymbol{y}_k^c \cdot \boldsymbol{y}_b^j\right)\right]} \tag{2}$$

$$\mathcal{L}_{clu} = - \log \frac{\sum\limits_{i=1}^{N} \exp\left(\boldsymbol{y}_{i^a,pos} \cdot \boldsymbol{y}_{i^b,pos} + \boldsymbol{y}_{i^a,neg} \cdot \boldsymbol{y}_{i^b,neg}\right)}{\sum\limits_{\forall (i,j,k,m)\in\Phi} \exp\left(\boldsymbol{y}_{i^k,pos} \cdot \boldsymbol{y}_{j^m,neg}\right)}, \tag{3}$$

Table 1: Notations of gradient derivations.

Notations	Dimensionality	Descriptions
N	$\in \mathbb{R}$	Mini-batch size
C	$\in \mathbb{R}$	Number of clusters
H	$\in \mathbb{R}$	Hidden dimension of cluster head g_C
a, b	$\in \mathbb{R}$	The first / second random augmentation
$\boldsymbol{w}^c$	$\in \mathbb{R}^H$	Weights of the last layer for cluster c
$\boldsymbol{H}_k$	$\in \mathbb{R}^{N \times H}$	Hidden feature matrix of a mini-batch using augmentation k
$\boldsymbol{H}_{k,pos}$, $\boldsymbol{H}_{k,neg}$	$\in \mathbb{R}^{N \times H}$	Positive / Negative feature matrix of a mini-batch using augmentation k
$\boldsymbol{h}_{i^k}$	$\in \mathbb{R}^H$	i th row of feature matrix $\boldsymbol{h}_k$
$\boldsymbol{h}_{i^k,pos}$, $\boldsymbol{h}_{i^k,neg}$	$\in \mathbb{R}^C$	i th row of positive / negative assignment matrix $\boldsymbol{h}_{k,pos}$ / $\boldsymbol{h}_{k,neg}$
$\boldsymbol{Y}_k$	$\in \mathbb{R}^{N \times C}$	Assignment matrix of a mini-batch using augmentation k
$\boldsymbol{Y}_{k,pos}$, $Y_{k,neg}$	$\in \mathbb{R}^{N \times C}$	Positive / Negative assignment matrix of a mini-batch using augmentation k
$\boldsymbol{y}_k^c$	$\in \mathbb{R}^N$	c th column of assignment matrix $\boldsymbol{Y}_k$
$\boldsymbol{y}_{i^k}$	$\in \mathbb{R}^C$	i th row of assignment matrix $\boldsymbol{Y}_k$
$\boldsymbol{y}_{i^k,pos}$, $\boldsymbol{y}_{i^k,neg}$	$\in \mathbb{R}^C$	i th row of positive / negative assignment matrix $\boldsymbol{Y}_{k,pos}$ / $\boldsymbol{Y}_{k,neg}$
$y^c_{i^k,pos}$, $y^c_{i^k,neg}$	$\in \mathbb{R}$	c th element of positive / negative assignment $\boldsymbol{y}_{i^k,pos}$ / $\boldsymbol{y}_{i^k,neg}$

where

$$\Phi = \{(i, j, k, m) \mid \forall\, i, j \in \{1, .., N\} \; and \, \forall\, k, m \in \{a, b\}\}.$$

B.1 Cluster-based: Cluster-level Contrastive

Recall that contrastive losses is to maximize the similarity between positive pairs and minimize the similarity of negative ones, For convenience, we consider pairwise relations and divide the gradient into two parts: (1) positive-pair term and (2) negative-pair term.

We start from discussing the stabilization of model generalization, which means the model could produce similar results no matter whatever kind of augmentation an image has been applied. Both CC [1], which proposed cluster-based clustering loss, and our model leverage a dual contrastive framework. Such dual contrastive framework incorporates instance-level contrastive learning, so high similarities between the instances augmented from the same image are encouraged. As a result, we have the assumption of generalization below.

Assumption 1 (Generalization) *Let a, b be any two random augmentations. The statements below hold:*

$$\begin{cases} \boldsymbol{y}_a^j \approx \boldsymbol{y}_b^j \approx \hat{\boldsymbol{y}}^j \quad \textit{for} \quad j \in \{1, ..., C\} \\ \boldsymbol{H}_a \approx \boldsymbol{H}_b \approx \hat{\boldsymbol{H}} \end{cases}$$

There is another assumption that helps our derivation. We assume the cluster head would produce confident results, which means there exits a high prediction score with respect to some specific cluster c'. To verify this assumption, we study on the confidence of our cluster assignment in Sec. C.1.

Assumption 2 (Confident Cluster Assignment) *There exists some $c' \in \{1, ..., C\}$ such that*

$$y_i^{c'} \approx 1 \quad \textit{and} \quad y_i^j \approx 0, \, \forall j \neq c'$$

This implies that

$$\boldsymbol{y}_a^c \cdot \boldsymbol{y}_a^j \approx \boldsymbol{y}_a^c \cdot \boldsymbol{y}_b^j \approx 0 \quad \text{for} \quad j \neq c, \; j \in \{1, ..., C\} \tag{4}$$

With the approximations above, the gradients of cluster-level clustering loss in terms of positive/negative pairs can be derived as:

$$
\begin{aligned}
\left.\frac{\partial \mathcal{L}^{clu}_{clu}}{\partial \boldsymbol{w}^c}\right|_{pos} &= -2\,\frac{\partial}{\partial \boldsymbol{w}^c}\log\left[\,\exp\left(\boldsymbol{y}^c_a\cdot\boldsymbol{y}^c_b\right)\right] \\
&\approx -2\,\frac{\partial}{\partial \boldsymbol{w}^c}\left(\hat{\boldsymbol{y}}^c\cdot\hat{\boldsymbol{y}}^c\right) \qquad \text{(with Asm. 1)} \\
&= -4\,\hat{\boldsymbol{H}}^{\mathsf{T}}\hat{\boldsymbol{y}}^c
\end{aligned} \tag{5}
$$

$$
\begin{aligned}
\left.\frac{\partial \mathcal{L}^{clu}_{clu}}{\partial \boldsymbol{w}^c}\right|_{neg} &= 2\,\frac{\partial}{\partial \boldsymbol{w}^c}\log\left\{\sum_{\substack{j=1\\ k\in\{a,b\}}}^{C}\mathbb{1}_{[j\neq c]}\left[\exp\left(\boldsymbol{y}^c_k\cdot\boldsymbol{y}^j_a\right)+\exp\left(\boldsymbol{y}^c_k\cdot\boldsymbol{y}^j_b\right)\right]\right\} \\
&= 2\,\frac{\partial}{\partial \boldsymbol{w}^c}\log\left\{\sum_{j=1}^{C}\mathbb{1}_{[i\neq c]}\left[\exp\left(\boldsymbol{y}^c_a\cdot\boldsymbol{y}^j_a\right)+\exp\left(\boldsymbol{y}^c_b\cdot\boldsymbol{y}^j_b\right)+2\ \exp\left(\boldsymbol{y}^c_a\cdot\boldsymbol{y}^j_b\right)\right]\right\} \\
&\approx 2\,\frac{\partial}{\partial \boldsymbol{w}^c}\log\left\{4\sum_{j=1}^{C}\mathbb{1}_{[j\neq c]}\exp\left(\hat{\boldsymbol{y}}^c\cdot\hat{\boldsymbol{y}}^j\right)\right\} \quad \text{(with Asm. 1)} \\
&= \frac{8\sum_{j=1}^{C}\left[\mathbb{1}_{[j\neq c]}\ \hat{\boldsymbol{H}}^{\mathsf{T}}\hat{\boldsymbol{y}}^j\right]}{4\sum_{j=1}^{C}\mathbb{1}_{[j\neq c]}\exp\left(\hat{\boldsymbol{y}}^c\cdot\hat{\boldsymbol{y}}^j\right)} \\
&\approx \frac{2}{C-1}\sum_{j=1}^{C}\left[\mathbb{1}_{[j\neq c]}\ \hat{\boldsymbol{H}}^{\mathsf{T}}\hat{\boldsymbol{y}}^j\right] \quad \text{(with Eq. 4)}
\end{aligned} \tag{6}
$$

Sum over Eq. 5 and Eq. 6, the gradient of cluster-based contrastive clustering loss is written as:

$$
\begin{aligned}
\frac{\partial \mathcal{L}^{clu}_{clu}}{\partial \boldsymbol{w}^c} &= \left.\frac{\partial \mathcal{L}^{clu}_{clu}}{\partial \boldsymbol{w}^c}\right|_{pos} + \left.\frac{\partial \mathcal{L}^{clus}}{\partial \boldsymbol{w}^c}\right|_{neg} \\
&= -4\,\hat{\boldsymbol{H}}^{\mathsf{T}}\hat{\boldsymbol{y}}^c + \frac{2}{C-1}\sum_{j=1}^{C}\left[\mathbb{1}_{[j\neq c]}\ \hat{\boldsymbol{H}}^{\mathsf{T}}\hat{\boldsymbol{y}}^j\right] \\
&= -2\,\hat{\boldsymbol{H}}^{\mathsf{T}}\left[2\,\hat{\boldsymbol{y}}^c - \frac{2}{C-1}\sum_{j=1}^{C}\mathbb{1}_{[j\neq c]}\ \hat{\boldsymbol{y}}^j\right]
\end{aligned} \tag{7}
$$

Discussion. The classifiers $\boldsymbol{w}^c$ is updated through $\boldsymbol{w}^c = \boldsymbol{w}^c - \eta\,\frac{\partial \mathcal{L}^{clu}_{clu}}{\partial \boldsymbol{w}^c}$ every step. Such optimization would pull each classifier $\boldsymbol{w}^c$ closer to the samples that are assigned to cluster c and push it away from the other samples. However, if there exists a cluster c' such that most samples in a mini-batch are assigned to cluster c', we can make the following observations. First, the classifier $\boldsymbol{w}^{c'}$ would get closer to most samples in a mini-batch. Second, for the other classifiers $\boldsymbol{w}^c, c \neq c'$, the strength of pushing classifiers away from most samples in a mini-batch (which are assigned to cluster c') is larger than the one that pulls them closer to the samples belonging to their corresponding cluster c. The observations above imply that the dominant cluster c' is becoming larger while the growth of the other clusters are being suppressed. Hence, the optimization through the cluster-based contrastive loss could not alleviate cluster degeneracy.

B.2 Instance-based: Cluster-level Contrastive Loss

Notice that we construct the communities in the instance-level subspace, where instance distinction is preserved with the guidance of instance-level contrastive loss. As we have discussed in appx. B.1, the assumption of model generalization holds, thus implying the statements in the following also hold:

$$\begin{cases} \boldsymbol{y}_{i^a,*} \approx \boldsymbol{y}_{i^b,*} \approx \hat{\boldsymbol{y}}_{i,*} \\ \boldsymbol{H}_{a,*} \approx \boldsymbol{H}_{b,*} \approx \hat{\boldsymbol{H}}_{*} \end{cases} \text{for} \quad i \in \{1, ..., N\},\ * \in \{pos, neg\} \tag{8}$$

With Asm. 2, we can further have

$$\boldsymbol{y}_{i^a,pos} \cdot \boldsymbol{y}_{i^b,pos} \approx \boldsymbol{y}_{i^a,neg} \cdot \boldsymbol{y}_{i^b,neg} \approx 1 \tag{9}$$

Hence, the gradients with regards to positive/negative pairs are:

$$\begin{aligned} \frac{\partial \mathcal{L}_{clu}}{\partial \boldsymbol{w}^c}\bigg|_{pos} &= -\frac{\partial}{\partial \boldsymbol{w}^c} \log \left\{ \sum_{i=1}^{N} \exp\left(\boldsymbol{y}_{i^a,pos} \cdot \boldsymbol{y}_{i^b,pos} + \boldsymbol{y}_{i^a,neg} \cdot \boldsymbol{y}_{i^b,neg}\right) \right\} \\ &\approx -\frac{\partial}{\partial \boldsymbol{w}^c} \log \left\{ \sum_{i=1}^{N} \exp\left(\hat{\boldsymbol{y}}_{i,pos} \cdot \hat{\boldsymbol{y}}_{i,pos} + \hat{\boldsymbol{y}}_{i^a,neg} \cdot \hat{\boldsymbol{y}}_{i^b,neg}\right) \right\} \qquad \text{(with Eq. 8)} \\ &\approx \frac{-2 \exp 2 \left[\hat{\boldsymbol{H}}_{pos}^{\intercal} \hat{\boldsymbol{y}}_{pos}^{c} + \hat{\boldsymbol{H}}_{neg}^{\intercal} \hat{\boldsymbol{y}}_{neg}^{c}\right]}{N \exp 2} \qquad \text{(with Eq. 9)} \\ &= -\frac{2}{N} \left[\hat{\boldsymbol{H}}_{pos}^{\intercal} \hat{\boldsymbol{y}}_{pos}^{c} + \hat{\boldsymbol{H}}_{neg}^{\intercal} \hat{\boldsymbol{y}}_{neg}^{c}\right] \end{aligned} \tag{10}$$

$$\begin{aligned} \frac{\partial \mathcal{L}_{clu}}{\partial \boldsymbol{w}^c}\bigg|_{neg} &= \frac{\partial}{\partial \boldsymbol{w}^c} \log \left\{ \sum_{\forall (i,j,k,m) \in \Phi} \exp\left(\boldsymbol{y}_{i^k,pos} \cdot \boldsymbol{y}_{j^m,neg}\right) \right\} \\ &\approx \frac{\partial}{\partial \boldsymbol{w}^c} \log \left\{ 4 \sum_{i=1}^{N} \sum_{j=1}^{N} \exp\left(\hat{\boldsymbol{y}}_{i,pos} \cdot \hat{\boldsymbol{y}}_{j,neg}\right) \right\} \qquad \text{(with Eq. 8)} \\ &= \frac{1}{\Psi} \left[\left(\hat{\boldsymbol{E}} \hat{\boldsymbol{H}}_{pos}\right)^{\intercal} \hat{\boldsymbol{y}}_{neg}^{c} + \left(\hat{\boldsymbol{E}} \hat{\boldsymbol{H}}_{neg}\right)^{\intercal} \hat{\boldsymbol{y}}_{pos}^{c} \right] \end{aligned} \tag{11}$$

where

$$\Psi \equiv \sum_{i=1}^{N} \sum_{j=1}^{N} \exp\left(\hat{\boldsymbol{y}}_{i,pos} \cdot \hat{\boldsymbol{y}}_{j,neg}\right) \tag{12}$$

$$\hat{\boldsymbol{E}}_{ij} \equiv \exp\left(\hat{\boldsymbol{y}}_{i,pos} \cdot \hat{\boldsymbol{y}}_{j,neg}\right), \hat{\boldsymbol{E}} \in \mathbb{R}^{N \times N} \tag{13}$$

Therefore, the total gradient of the proposed instance-based contrastive loss is derived as:

$$\begin{aligned} \frac{\partial \mathcal{L}_{clu}}{\partial \boldsymbol{w}^c} &= \frac{\partial \mathcal{L}_{clu}}{\partial \boldsymbol{w}^c}\bigg|_{pos} + \frac{\partial \mathcal{L}_{clu}}{\partial \boldsymbol{w}^c}\bigg|_{neg} \\ &= -\frac{2}{N} \left[\hat{\boldsymbol{H}}_{pos}^{\intercal} \hat{\boldsymbol{y}}_{pos}^{c} + \hat{\boldsymbol{H}}_{neg}^{\intercal} \hat{\boldsymbol{y}}_{neg}^{c}\right] + \frac{1}{\Psi} \left[\left(\hat{\boldsymbol{E}} \hat{\boldsymbol{H}}_{pos}\right)^{\intercal} \hat{\boldsymbol{y}}_{neg}^{c} + \left(\hat{\boldsymbol{E}} \hat{\boldsymbol{H}}_{neg}\right)^{\intercal} \hat{\boldsymbol{y}}_{pos}^{c} \right] \end{aligned} \tag{14}$$

C More Ablation Studies

C.1 Confidence Learning Curve

We plot the confidence learning curve by averaging the maximum assignment score of each sample within a mini-batch. As shown Fig. 2, the confidence score are quite high (approximate to 1) after a few training epochs, which verifies the assumption we made for the gradient derivation.

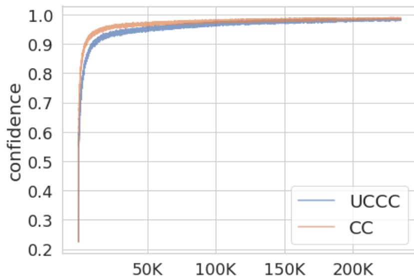


Figure 2: The learning curve of assignment confidence on CIFAR-10.

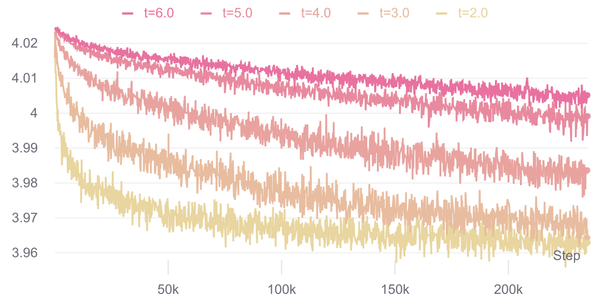


Figure 3: The training curves of our proposed cluster-level contrastive loss using different values of the tolerant parameter t on CIFAR-10.

C.2 Tolerant Parameter

To observe the influence of the tolerant parameter t for positive confidence threshold, we conduct experiments varying t from $\{2,3,4,5,6\}$. Table 2 shows the proposed model achieves the best performance when $t = 4$. Additionally, if the tolerant parameter is too large, the performances on three considered metrics might drop significantly. As the large t implies the low threshold value to construct positive peer relations, information from the noisy peers may cause the model hard to distinguish samples from different clusters.

Table 2: The influence of different tolerant parameter t on CIFAR-10 dataset.

Metrics	NMI	ARI	ACC
$t = 2$	70.8	64.6	79.0
$t = 3$	73.1	70.1	83.0
$t = 4$	**74.8**	**71.0**	**84.2**
$t = 5$	74.0	70.4	83.8
$t = 6$	69.9	64.4	77.8

According to the loss curves (Fig. 3), we also find that larger values of t usually require more training steps for model convergence.

C.3 Qualitative Analysis

We conducted a qualitative analysis to examine how well the clustering result is on CIFAR-10, ImageNet-10, and large-scale tiny-ImageNet dataset.

CIFAR-10 Dataset. The confusion matrix between the ground truth labels and classification results is sown in Fig. 4. A perfect classification would only place items on the diagonal line. The figure shows that the proposed model finds the right cluster for most images except for some error-prone classes such as *birds*, *cats*, and *dogs*.

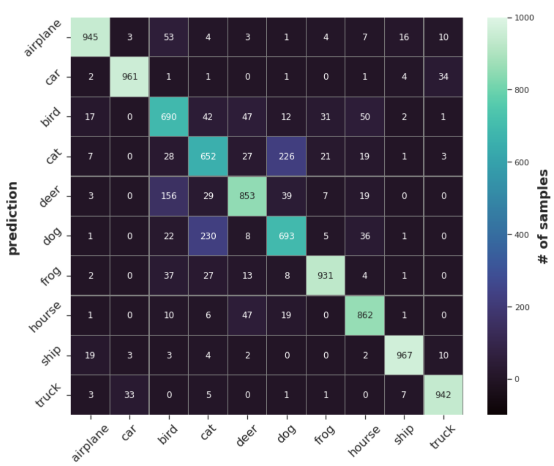
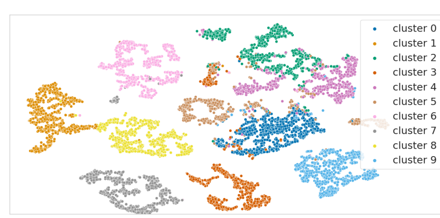
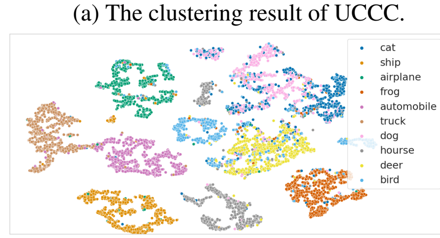


Figure 4: The confusion matrix on CIFAR-10.



(a) The clustering result of UCCC.



(b) Ground-truth labels.

Figure 5: t-SNE visualization on CIFAR-10.

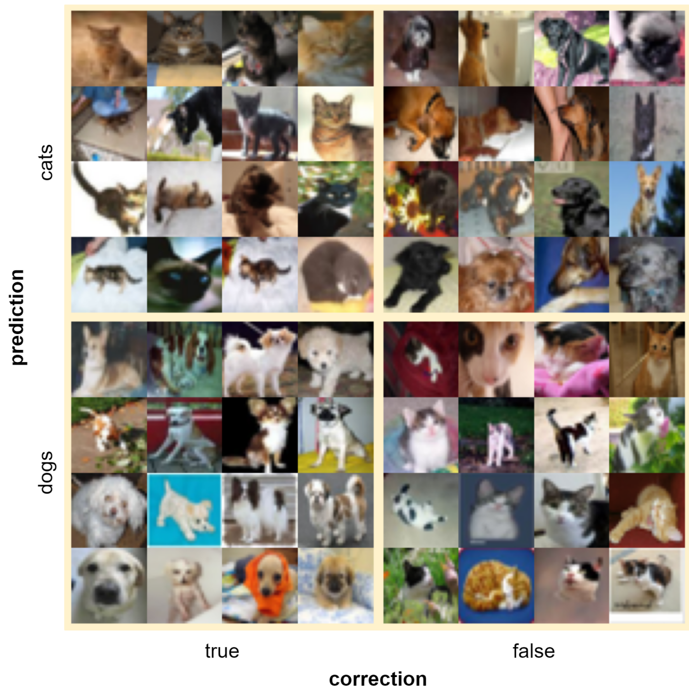


Figure 6: Example images from the cluster *dogs* and *cats* on CIFAR-10. The two left blocks contain the correctly classified images and those misclassified images are demonstrated in the right blocks.

We further perform t-SNE visualization in the instance-level embedding space. Fig. 5 compares the visualization results of predicted clusters and the ground-truth labels, where different colors indicated different labels/clusters. As shown in Fig. 5b, it is observed that the extracted features from class *dogs* and class *cat* (denoted in navy and light pink respectively) are hardly distinguishable. This explains why many dog images are misclassified as cats, and vice versa (Fig. 6). Nonetheless, the result in Fig. 5a still proves the efficacy of the instance-level contrastive learning since the features from different clusters are mostly well separated.

ImageNet-10 Dataset. The confusion matrix and the t-SNE visualization are demonstrated in the figures below. In Fig. 7, a high concentration of items on the diagonal line confirms the proposed model correctly groups all samples into 10 classes. Fig. 8 also verifies our clustering result is almost the same as the ground-truth labels.

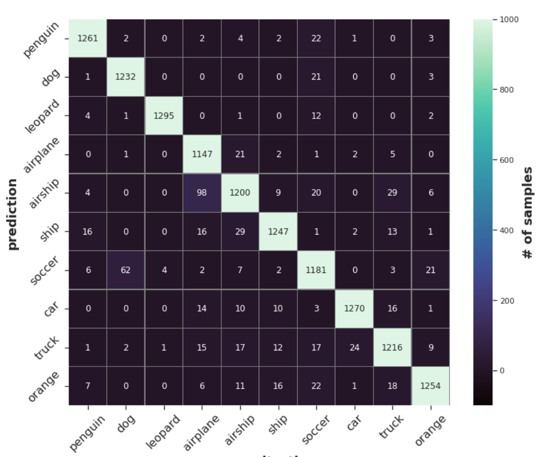
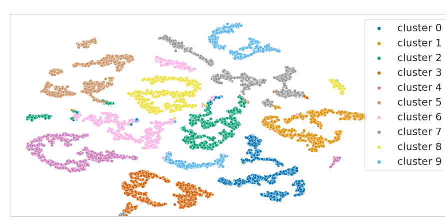


Figure 7: The confusion matrix on ImageNet-10.



(a) The clustering result of UCCC.

(b) Ground-truth labels.

Figure 8: t-SNE visualization on ImageNet-10.

Superclass	Classes				
Animals	goldfish	salamandra	bullfrog	bell toad	American alligator
	boa constrictor	trilobite	scorpion	black widow	tarantula
	centipede	goose	koala	jellyfish	bruin
	brain coral	snail	slug	sea slug	Maine lobster
	crawfish	black stork	king penguin	albatross	dugong
	Chihuahua	Yorkshire terrier	golden retriever	alsatian	standard poodle
	tabby	Persian cat	Egyptian cat	puma	lion
Insects	ladybug	fly	bee	hopper	walkingstick
	roach	mantis	dragonfly	monarch	sulfur butterfly
Others	holothurian	guinea pig	hog	ox	bison
	bighorn	gazelle	dromedary	orang	chimp
	baboon	African elephant	panda	abacus	judge's robe
	altar	apron	backpack	banister	barbershop
	barn	cask	basketball	tub	wagon
	beacon	beaker	beer bottle	bikini	binoculars
	birdhouse	bowtie	brass	broom	pail
	bullet	meat market	taper	cannon	cardigan
	ATM	CD player	chain	chest	Christmas stoching
	cliff dwelling	keypad	candy store	convertible	crane
	dam	desk	board	drumstick	dumbbell
	flagpole	fountain	freight car	frypan	fur coat
	gasmask	go-kart	gondola	hourglass	iPos
	ricksha	kimono	lampshade	mower	lifeboat
	limo	magnetic compass	maypole	military uniform	mini
	moving van	nail	neck brace	obelist	oboe
	organ	parking meter	pay-phone	paling	pill bottle
	plunger	pole	wagon	poncho	pop bottle
	potter's wheel	missile	punch bag	reel	icebox
	remote	rocker	rugby ball	sandal	school bus
	scoreboard	sewing machine	snorkel	sock	sombrero
	space heater	spider web	sport car	steel arch bridge	stopwatch
	shades	suspension bridge	bathing trunks	syringe	teapot
	teddy	thatch	torch	tractor	triumphal arch
	trolleybus	turnstile	umbrella	vestment	viaduct
	volleyball	water jug	water tower	wok	wooden spoon
	drop	coral reef	lakeside	coast	acorn
	comic book	plate	alp		
Food	guacamole	icecream	lolly	pretzel	mashed potato
	cauliflower	bell pepper	mushroom	orange	lemon
	banana	poegranate	meatloaf	pizza	potpie
	espresso				

Table 3: Superclass definition for tiny-ImageNet.

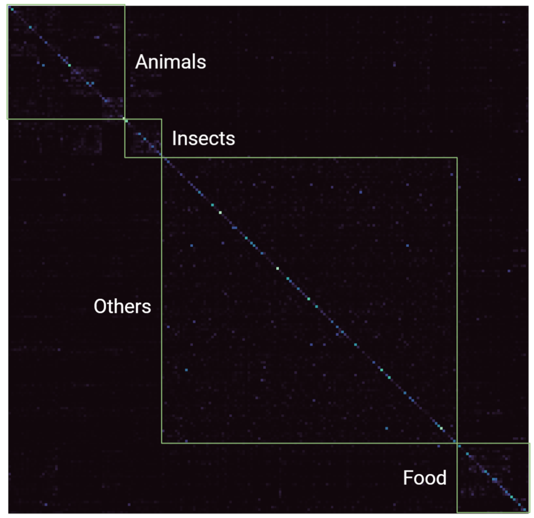


Figure 9: The confusion matrix on tiny-ImageNet.

Tiny-ImageNet Dataset - 200 classes. In Fig. 9, we mark the superclasses defined in Table 3. The results show that the misclassified examples tend to be assigned to other clusters from within the same superclass. Additionally, we demonstrate images from the testing set that were assigned to the same cluster in Fig. 10, 11, and 12. In particular, Fig. 12 shows some failure cases where different objects are grouped together due to similar image background.

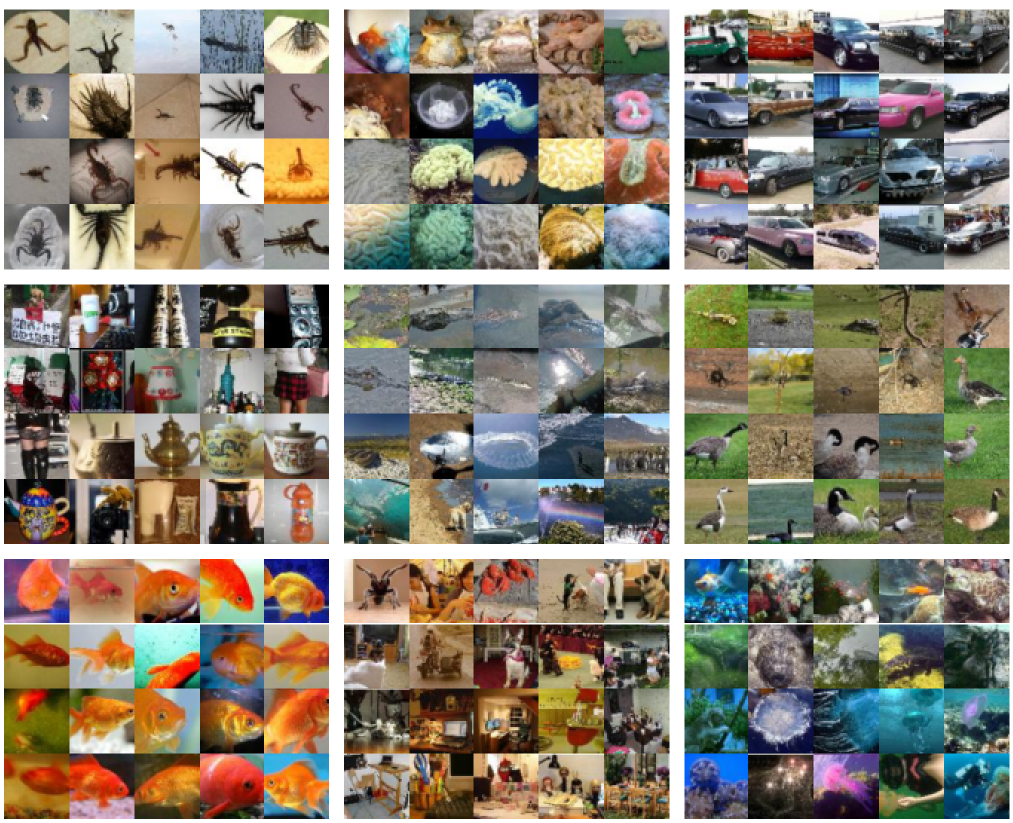


Figure 10: Example clusters of tiny-ImageNet (1).

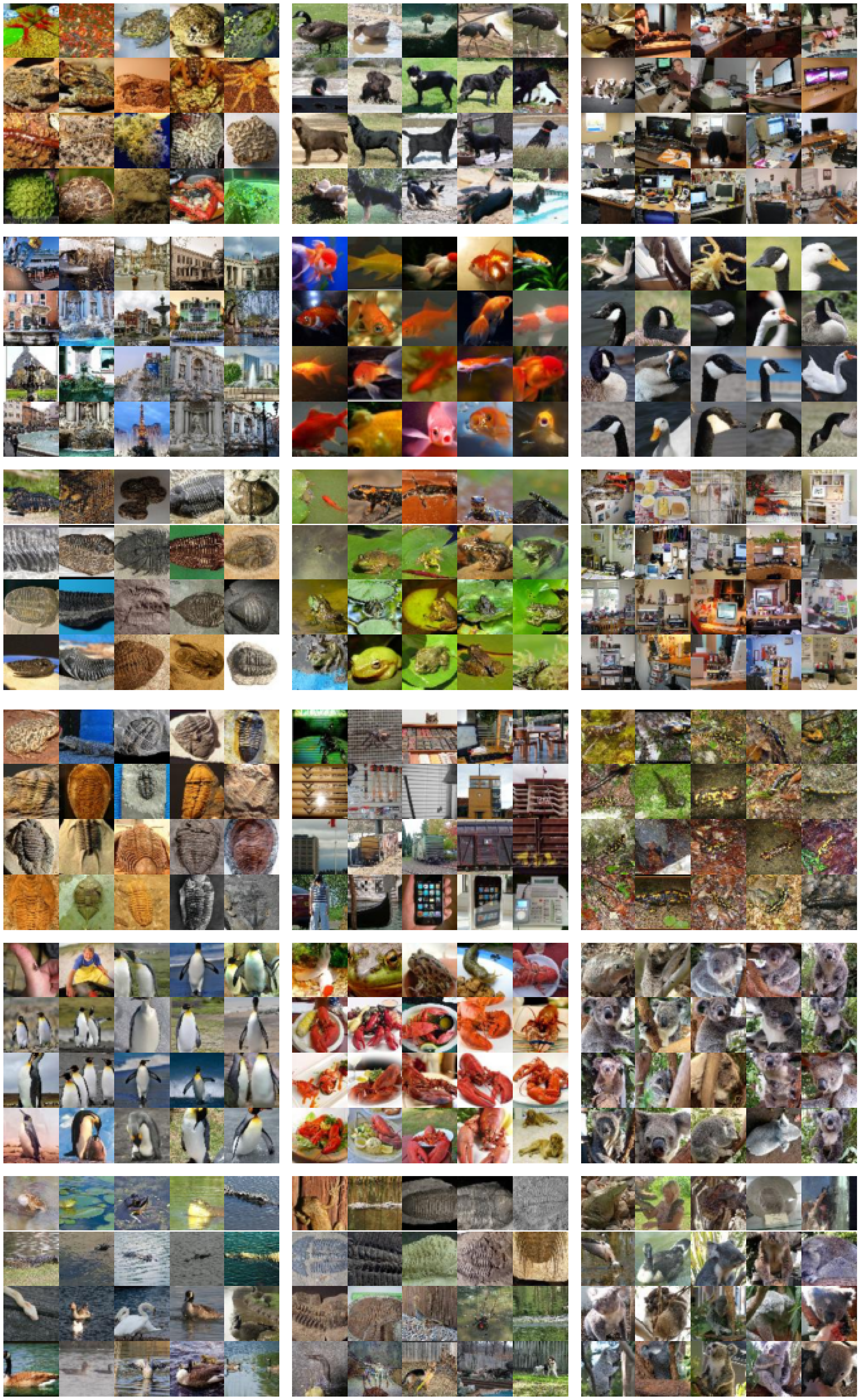


Figure 11: Example clusters of tiny-ImageNet (2).

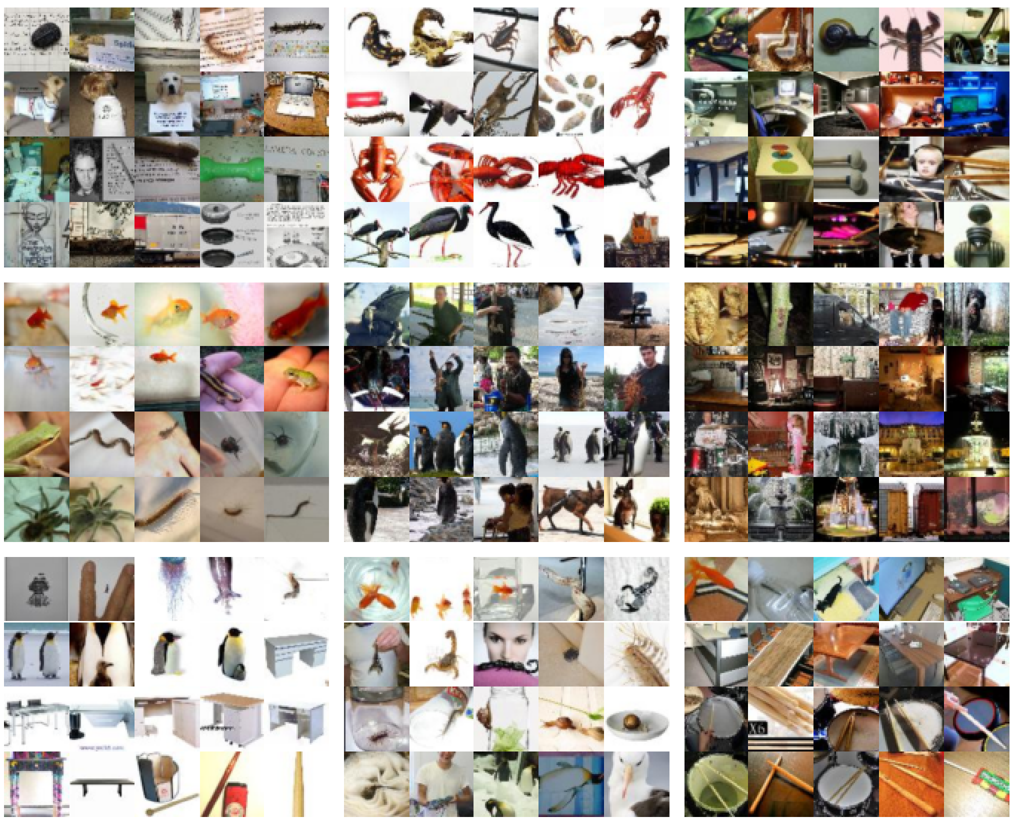


Figure 12: Incorrect clusters of tiny-ImageNet predicted by our model.

References

[1] Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. Contrastive clustering. *arXiv preprint arXiv:2009.09687*, 2020.