UNSUPERVISED MANIFOLD LINEARIZING AND CLUSTERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Clustering data lying close to a union of low-dimensional manifolds, with each manifold as a cluster, is a fundamental problem in machine learning. When the manifolds are assumed to be linear subspaces, many methods succeed using low-rank and sparse priors, which have been studied extensively over the past two decades. Unfortunately, most real-world datasets can not be well approximated by linear subspaces. On the other hand, several works have proposed to identify the manifolds by learning a feature map such that the data transformed by the map lie in a union of linear subspaces, even though the original data are from non-linear manifolds. However, most works either assume knowledge of the membership of samples to clusters, or are shown to learn trivial representations. In this paper, we propose to simultaneously perform clustering and learn a union-of-subspace representation via Maximal Coding Rate Reduction. Experiments on synthetic and realistic datasets show that the proposed method achieves clustering accuracy comparable with state-of-the-art alternatives, while being more scalable and learning geometrically meaningful representations.

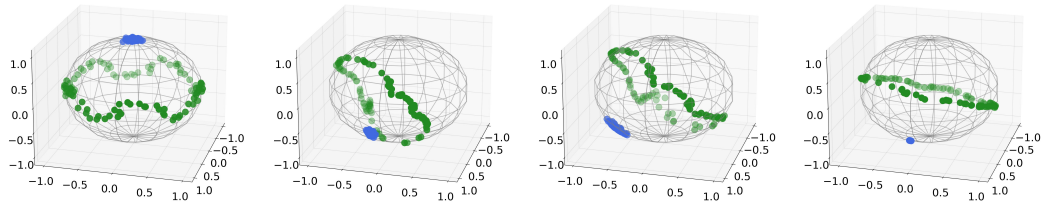
1 INTRODUCTION

1.1 MOTIVATION AND CONTRIBUTIONS

Clustering is a fundamental problem in machine learning, allowing one to group data into clusters based on assumptions about the geometry of clusters. For example, when data are concentrated around distinct centroids, classical k-means clustering (Lloyd, 1957; Forgey, 1965; Jancey, 1966; MacQueen, 1967) and its variants (Bradley et al., 1996; Arthur & Vassilvitskii, 2006; Bahmani et al., 2012) are able to find the cluster centroids and assign membership to each data point. More generally[1], subspace clustering methods (Elhamifar & Vidal, 2009; 2013; Lu et al., 2012; Liu et al., 2013; Heckel & Bölcskei, 2015; You et al., 2016a) are designed to cluster data that lie close to a union of different low-dimensional linear (or affine) subspaces, where each subspace defines a cluster. Overall, those methods often enjoy theoretical guarantees of correct clustering (Soltanolkotabi & Candés, 2012; Lu et al., 2012; Elhamifar & Vidal, 2013; Soltanolkotabi et al., 2014; Wang et al., 2015; Wang & Xu, 2016; You et al., 2016b;a; Tsakiris & Vidal, 2018; Li et al., 2018; You et al., 2019; Robinson et al., 2019) and find applications in various problems such as image clustering, face recognition, motion segmentation, and recently in popular Transformer architectures in deep learning (Roy et al., 2021).

Despite the wide range of applications and theoretical guarantees, subspace clustering methods rely on a crucial assumption that each cluster can be well approximated by a linear/affine subspace, which is often not valid for many real-world datasets. For instance, even in a dataset as simple as MNIST hand-written digits, images of a single digit do not lie close to a low-dimensional linear subspace, thus directly applying subspace clustering will fail. Instead, it is more a natural to assume the clusters are from non-linear low-dimensional manifolds (one manifold per cluster), and attempt to learn or design a *non-linear embedding* of the data so that the transformed data lies close to distinct linear subspaces, with points from one manifold mapped to the same subspace. For example, Lim et al. (2020) shows that a subspace clustering method can achieve 99% clustering accuracy on MNIST images after embedding the data with the scattering transform Bruna & Mallat (2013).

[1]This includes k-means-based methods, since a centroid is a 0-dimensional affine subspace.



(a) (b) (c) (d)

Figure 1: (a) Input data $\boldsymbol{X}$ of two manifolds each containing 100 points. (b) Features $\boldsymbol{Z}_{\boldsymbol{\theta}}$ at random initialization. (c) $\boldsymbol{Z}_{\boldsymbol{\theta}}$ after self-supervised initialization. (d) $\boldsymbol{Z}_{\boldsymbol{\theta}}$ after MLC (4) training. Details are in Appendix B.

Beyond the above example, numerous other subspace clustering methods have explored hand-designing an appropriate feature embedding (or kernel) such as polynomial or exponential mappings (Patel & Vidal, 2014). However, these embeddings assume specific families of manifolds, thus they need to be hand-crafted for various tasks and datasets using domain knowledge, which makes their application challenging for complicated data such as natural images. On the other hand, Elhamifar & Vidal (2011) proposes to cluster data based on treating a local neighborhood of the manifold approximately as a linear subspace. However, for this to succeed sufficient sampling density is required, which implies a prohibitive number of samples when the manifolds are of moderate dimension or are highly curved. Further, for a new sample unseen at training time one needs to run the algorithm with all samples to embed it or assign a membership to it, which is expensive computationally. More recently, numerous works propose to learn an appropriate linear embedding of the data via deep networks and then perform subspace clustering in the feature space (Peng et al., 2017; Ji et al., 2017; Abavisani & Patel, 2018; Zhang et al., 2019; Kheirandishfard et al., 2020). Unfortunately, it has been shown that many of these methods are provably ill-posed and learn trivial representations[2], with much of the claimed benefit coming from ad-hoc post-processing rather than the method itself (Haeffele et al., 2020). This leads to the following question:

Question 1. *For data approximately supported on an underlying union of manifolds, can we learn a transformation of the data, so that the transformed data lie in distinct linear subspaces to be easily clustered?*

Meanwhile, learning a representation from multi-modal data has been a topic of its own interest in machine learning. An ideal property of the learned representation often pursued is *between-cluster discrimination*, i.e., features from different clusters should be well separated. Further, an important yet often ignored property of the learned representation is that it maintains *within-cluster diversity*. This is desirable as it allows distances of samples within a cluster to be preserved under the learned transformation, which could facilitate downstream tasks such as denoising, hierarchical clustering and semantic interpretation. In the supervised setting, training with the cross-entropy (CE) classification objective fails to achieve the second property, as it has been shown empirically (Papyan et al., 2020) and theoretically (Tirer & Bruna, 2022; Zhou et al., 2022) that the representation learned by CE has the property that features from one cluster tend to collapse to a single point. On the other hand, recent work has proposed the principle of Maximal Coding Rate Reduction (MCR^2) Yu et al. (2020) as one of the few methods that are able to achieve the two ideal properties by learning a representation where features from each cluster are expected to lie close to a low-dimensional subspace (within-cluster diverse), and the subspaces from different clusters are orthogonal to each other (between-cluster discriminative). However, for MCR^2 to learn such orthogonal subspaces each corresponding to one cluster, one needs the annotation of which sample belong to which cluster. Such annotation might be expensive or impossible to acquire for large-scale datasets. This motivates another question of interest:

Question 2. *Can we learn a union-of-orthogonal-subspace representation of data coming from an underlying union of manifolds without access to the labels?*

This paper gives positive answers to the two interrelated questions by contributing the following.

[2]In this paper, we use 'representation' and 'feature' interchangeably to mean the image of data under a (learned) transformation.

1. We propose to simultaneously cluster the data and learn a union-of-orthogonal-subspace representation via MCR^2, when data is assumed to lie close to a union of manifolds. This is achieved by formulation (4), which optimizes over both the representation and a *doubly stochastic* cluster membership formulation inspired by the state-of-the-art subspace clustering result Lim et al. (2020).

2. Since the membership has as many entries as the square of the batch size of the input data, we give a parameterization of the membership (Figure 2). Further, as problem (4) is highly non-convex, we give a meta-algorithm (Algorithm 1) on how to initialize the variables and to optimize it.

3. We conduct experiments on simulation and CIFAR10 to demonstrate some desirable properties of the proposed method. We further experiment on datasets with larger number of clusters and imbalanced clusters such as CIFAR100-20, CIFAR100-100, and TinyImageNet200, and show that the proposed method achieves state-of-the-art performance.

1.2 Additional Related Work

Beyond the above, we make connections to a few important works that are related to this paper.

Deep Clustering and Representation Learning. Recently, there is an interesting line of research in representation learning and clustering that takes advantage of pseudo-labelling and semi/self-supervised learning (Caron et al., 2018; Van Gansbeke et al., 2020; Park et al., 2021; Niu et al., 2021). Specifically, one first identifies a subset of samples (often termed reliable samples) based on geometric or statistical criteria in the learned representation and cluster prediction, and then uses the predicted labels for those reliable samples as if they are ground-truth labels to refine the representation and cluster prediction of other samples. Despite the promising clustering performance, the representation learned by these methods are not constrained to be both between-cluster discriminative and within-cluster diverse. In contrast, the proposed method learns a representation with these two ideal properties (see Figure 4) and also achieves state-of-the-art clustering performance (see Tables 2 and 4).

Neural Manifold Clustering and Embedding (NMCE). A recent preprint (Li et al., 2022) also proposes a solution to the same problem we study, i.e., clustering the data and learning an union-of-orthogonal-subspace representation. In particular, NMCE proposes to model the *point-to-cluster* membership and optimize MCR^2 (Yu et al., 2020) over both the representation and the membership. In this paper, we adopt a similar formulation, but we propose to model the *point-to-point* affinity using a doubly stochastic matrix, inspired by the state-of-the-art subspace clustering methods (§2.2). Aside from having different conceptual formulations and algorithms, our formulation is much more stable with respect to initialization and is naturally suitable for hierarchical clustering. We detail these distinctions in §2.2. Experiments (Table 2) further show that the proposed method (MLC) achieves higher accuracy than NMCE on large scale realistic datasets.

2 Problem Formulation

We start by defining the problem that we study. Suppose $\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_n] \in \mathbb{R}^{D \times n}$ is a dataset of n samples drawn from a union of k underlying manifolds $\bigcup_{j=1}^{k} \mathcal{M}_j$ and $\boldsymbol{y} \in \mathbb{R}^n$ their memberships to the manifolds, i.e., $\boldsymbol{x}_i \in \mathcal{M}_{y(i)}$.

Problem 1 (Unsupervised Manifold Linearizing and Clustering)**.** Given the dataset $\boldsymbol{X}$, can we simultaneously 1) *cluster the samples*, i.e., estimate $\boldsymbol{y}$, and 2) *learn a linear representation for manifolds*, i.e., find a transformation $f : \mathbb{R}^D \to \mathbb{R}^d$, such that the image of each manifold $f(\mathcal{M}_i)$ is a low-dimensional linear subspace of $\mathbb{R}^d$, and the subspaces satisfy desired properties (§1), i.e., they are between-cluster discriminative and within-cluster diverse?

Here we base our approach on the principle of Maximal Coding Rate Reduction (MCR^2) which is designed to learn ideal representations in the supervised case, i.e., when the membership $\boldsymbol{y}$ is given (§2.1). Then we discuss the challenges of simultaneously clustering and learning representation (Problem 1), and propose our MCR^2 clustering objective to solve Problem 1 with those challenges in mind (§2.2). We further give an algorithm to optimize the proposed objective (§2.3).

2.1 SUPERVISED MANIFOLD LINEARIZING VIA MCR^2

In the case when the labels $\boldsymbol{y}$ are given as supervision, MCR^2 Yu et al. (2020) aims to address part 2) of Problem 1. Let $f_{\boldsymbol{\theta}} : \mathbb{R}^D \to \mathbb{S}^{d-1}$ be a featurizer parameterized by a neural network, which in turn gives an embedding $\boldsymbol{Z}_{\boldsymbol{\theta}} := [\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n] \in \mathbb{R}^{d\times n}$ of data with $\boldsymbol{z}_i := f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) \in \mathbb{S}^{d-1}$. MCR^2 aims to learn an ideal representation by optimizing

$$\max_{\boldsymbol{\theta}} \quad \underbrace{\log\det\left(\boldsymbol{I} + \frac{d}{n\epsilon^2}\boldsymbol{Z}_{\boldsymbol{\theta}}\boldsymbol{Z}_{\boldsymbol{\theta}}^\top\right)}_{R(\boldsymbol{Z}_{\boldsymbol{\theta}};\,\epsilon)} - \underbrace{\sum_{j=1}^{k}\frac{\langle\boldsymbol{\Pi}_j,\boldsymbol{1}\rangle}{n}\log\det\left(\boldsymbol{I} + \frac{d}{\langle\boldsymbol{\Pi}_j,\boldsymbol{1}\rangle\epsilon^2}\boldsymbol{Z}_{\boldsymbol{\theta}}\,\mathrm{Diag}(\boldsymbol{\Pi}_j)\boldsymbol{Z}_{\boldsymbol{\theta}}^\top\right)}_{R_c(\boldsymbol{Z}_{\boldsymbol{\theta}},\boldsymbol{\Pi};\,\epsilon)},$$
$$\text{s.t.} \quad \boldsymbol{z}_i \in \mathbb{S}^{d-1}, \quad \forall i \in [n]. \tag{1}$$

where it is assumed that $f_{\boldsymbol{\theta}}$ is constrained to output unit ℓ_2 norm vectors in $\mathbb{R}^d$ (i.e., on the sphere $\mathbb{S}^{d-1}$), $\boldsymbol{\Pi} \in \mathbb{R}^{n\times k}$ is a given membership matrix such that $\Pi_{ij} = 1$ if $j = y(i)$ and $\Pi_{ij} = 0$ otherwise, $\epsilon > 0$ is a prescribed precision parameter, $\boldsymbol{\Pi}_j \in \mathbb{R}^n$ denotes the j^{th} column of $\boldsymbol{\Pi}$, $\boldsymbol{1}$ is a vector of all ones so that $\langle\boldsymbol{\Pi}_j, \boldsymbol{1}\rangle$ is the number of points in cluster j, and for $\boldsymbol{v} \in \mathbb{R}^n$, $\mathrm{Diag}(\boldsymbol{v})$ denotes a diagonal matrix with the entries of $\boldsymbol{v}$ along the diagonal.

Intuitively[3], the $R(\boldsymbol{Z}_{\boldsymbol{\theta}};\,\epsilon)$ term of (1) measures the volume of $\boldsymbol{Z}_{\boldsymbol{\theta}}$, and maximizing it would diversify features from all samples, which we will refer to as the *expansion term*. Likewise, the $R_c(\boldsymbol{Z}_{\boldsymbol{\theta}}, \boldsymbol{\Pi};\,\epsilon)$ term measures the sum of volumes of each class of $\boldsymbol{Z}_{\boldsymbol{\theta}}$ and is referred to as the *compression term*, since minimizing it would push features within each class to stay close. It has been shown that given $\boldsymbol{\Pi}$, the features obtained by maximizing the rate reduction $R(\boldsymbol{Z}_{\boldsymbol{\theta}};\,\epsilon) - R_c(\boldsymbol{Z}_{\boldsymbol{\theta}}, \boldsymbol{\Pi};\,\epsilon)$ has the property that the features of each cluster spread uniformly within a subspace (within-cluster diverse), and the subspaces from different clusters are orthogonal (between-cluster discriminative), under relatively mild assumptions (Yu et al., 2020).

2.2 UNSUPERVISED MANIFOLD LINEARIZING AND CLUSTERING VIA MCR^2

While the MCR^2 formulation is designed to learn ideal representations (§1) when the membership $\boldsymbol{y}$ (or equivalently $\boldsymbol{\Pi}$) is given, here we are interested in the unsupervised setting where one does not have access to membership annotations. Thus, we propose to simultaneously perform both parts 1) and 2) of Problem 1 by also optimizing over the membership $\boldsymbol{\Pi}$ of each data point. This naturally leads to

$$\max_{\boldsymbol{\theta},\boldsymbol{\Pi}\in\Omega_\circ} \quad R(\boldsymbol{Z}_{\boldsymbol{\theta}};\,\epsilon) - R_c(\boldsymbol{Z}_{\boldsymbol{\theta}}, \boldsymbol{\Pi};\,\epsilon), \quad \text{s.t.} \quad \boldsymbol{z}_i \in \mathbb{S}^{d-1}, \quad \forall i \in [n], \tag{2}$$

where $\Omega_\circ := \{\boldsymbol{\Pi} \in \mathbb{R}^{n\times k} : \forall i \in [n],\ \exists \hat{y}(i) \quad \text{s.t. } \Pi_{i\hat{y}(i)} = 1 \text{ and } \Pi_{ij} = 0 \text{ for } j \neq \hat{y}(i)\}$ is the set of all 'hard' assignments, i.e., each row of $\boldsymbol{\Pi}$ is a one-hot vector. However, this optimization is in general combinatorial: its complexity grows exponentially in n and k, and it does not allow smooth and gradual changes of $\boldsymbol{\Pi}$. Further, a second challenge is the chicken-and-egg nature of this problem: If one already has an ideal representation $\boldsymbol{Z}$, then existing subspace clustering methods can be applied on $\boldsymbol{Z}$ to estimate the membership. Likewise, if one is given the membership $\boldsymbol{\Pi}$ of clusters, then solving (1) would lead to an ideal representation. However, the $\boldsymbol{Z}_{\boldsymbol{\theta}}$ and $\boldsymbol{\Pi}$ at the beginning of optimization is typically far from ideal.

Doubly Stochastic Subspace Clustering. To address the combinatorial of estimating the cluster memberships, we draw inspiration from the closely related problem of *subspace clustering*, where the goal is to cluster n samples assumed to lie close to a union of k low-dimensional subspaces (§1). In this case, one typically does not directly learn an $n \times k$ matrix denoting memberships of n points into k subspaces. Instead, one first learns an affinity matrix $\boldsymbol{\Pi} \in \mathbb{R}^{n\times n}$ signaling the similarity between pairs of points, and then applies spectral clustering on the learned $\boldsymbol{\Pi}$ to obtain a final clustering (Elhamifar & Vidal, 2009; 2013; Lu et al., 2012; Liu et al., 2013; Heckel & Bölcskei, 2015; You et al., 2016a). In particular, requiring doubly-stochastic constraints on the affinity $\boldsymbol{\Pi}$ is shown theoretically to suppress false inter-cluster connections for clustering problems (Ding et al.,

[3]More formally, terms of the form $\log\det\left(I + \frac{d}{n\epsilon^2}\boldsymbol{W}\boldsymbol{W}^\top\right)$ estimate the average number of bits needed to code n i.i.d. samples $\boldsymbol{W} \in \mathbb{R}^{d\times n}$ from a zero-mean d-dimensional Gaussian up to a distortion ϵ (Ma et al., 2007), hence the name coding rate.

2022) along with state-of-the-art empirical performance for subspace clustering problems Lim et al. (2020).

Inspired by the above, we propose a constraint set Ω for the matrix $\mathbf{\Pi}$ to be the set of $n \times n$ doubly stochastic matrices,

$$\Omega = \{\mathbf{\Pi} \in \mathbb{R}^{n \times n} : \mathbf{\Pi} \geq 0, \quad \mathbf{\Pi}\mathbf{1} = \mathbf{\Pi}^\top \mathbf{1} = \mathbf{1}\}. \tag{3}$$

However, this constraint alone is insufficient for strong clustering performance: Consider the optimization of (2) with respect to $\mathbf{\Pi} \in \Omega$ only, and note that the objective is strongly convex with respect to $\mathbf{\Pi}$. Since we maximize a convex function with respect to convex constraints Ω, an optimal $\mathbf{\Pi}$ would lie at an extreme point of Ω, which for doubly stochastic matrices is a permutation matrix. This is not ideal for clustering, as it implies that every point is assigned to its own distinct cluster, and there is no incentive to merge points into larger clusters. To resolve this issue, we follow the approach in Lim et al. (2020) and add ℓ_2 regularization[4] $\frac{\gamma}{2}\|\mathbf{\Pi}\|_F^2$ to $\mathbf{\Pi}$ which biases $\mathbf{\Pi}$ toward the uniform matrix $\frac{1}{n}\mathbf{1}\mathbf{1}^\top$, so by tuning γ we can also tune the sparsity level of $\mathbf{\Pi}$. This results in our final proposed formulation, dubbed Manifold Linearizing and Clustering (MLC):

$$\begin{aligned} \max_{\boldsymbol{\theta}} \quad & \underbrace{\log\det\left(\boldsymbol{I} + \frac{d}{n\epsilon^2}\boldsymbol{Z}_{\boldsymbol{\theta}}\boldsymbol{Z}_{\boldsymbol{\theta}}^\top\right)}_{R(\boldsymbol{Z}_{\boldsymbol{\theta}};\,\epsilon)} - \underbrace{\frac{1}{n}\sum_{j=1}^{n}\log\det\left(\boldsymbol{I} + \frac{d}{\epsilon^2}\boldsymbol{Z}_{\boldsymbol{\theta}}\operatorname{Diag}((\mathbf{\Pi}_{\boldsymbol{\theta}})_j)\boldsymbol{Z}_{\boldsymbol{\theta}}^\top\right)}_{R_c(\boldsymbol{Z}_{\boldsymbol{\theta}},\mathbf{\Pi}_{\boldsymbol{\theta}};\,\epsilon)} - \frac{\gamma}{2}\|\mathbf{\Pi}_{\boldsymbol{\theta}}\|_F^2, \\ \text{s.t.} \quad & \boldsymbol{z}_i \in \mathbb{S}^{d-1}, \quad \forall i \in [n]; \quad \mathbf{\Pi}_{\boldsymbol{\theta}} \in \Omega, \end{aligned} \tag{4}$$

where $\mathbf{\Pi}_{\boldsymbol{\theta}} = \mathbf{\Pi}_{\boldsymbol{\theta}}(\boldsymbol{X})$ is now also parameterized by a neural network. While this is constrained optimization which may appear difficult to handle, we explain in §2.3 how we parameterize $\boldsymbol{Z}_{\boldsymbol{\theta}}$ and $\mathbf{\Pi}_{\boldsymbol{\theta}}$ via neural networks so that the constraints are satisfied by construction. Below, we note a few advantages of the proposed formulation.

Parameterizing $\mathbf{\Pi}$ via a Neural Network versus Free Variables. An alternative way to parameterize the membership would be to directly take $\mathbf{\Pi}$ as decision variables in Ω, as opposed to outputs of a neural network. However, this leads to maintaining $O(n^2)$ variables which is prohibitive for large datasets (e.g., $n = 10^6$ for ImageNet). In contrast, this is not the case if one parameterizes $\mathbf{\Pi}$ as a neural network, since one can do stochastic gradient descent such that for each batch both the memory and computational complexity is at most square of the batch size (Figure 2).

Comparison with NMCE. As mentioned in §**??**, NMCE (Li et al., 2022) approaches Problem 1 also by optimizing MCR^2 over both the representation and membership. However, in NMCE the membership is parameterized by an $n \times k$ matrix $\mathbf{\Pi}_{n\times k}$ that models the *point-cluster* membership, which is different from our *doubly stochastic point-point* membership matrix $\mathbf{\Pi}_{\boldsymbol{\theta}}$ inspired from the state-of-the-art subspace clustering. Note further that for NMCE the initialization of $\mathbf{\Pi}_{n\times k}$ is arbitrary and has nothing to do with the structures in the initialized representation $\mathbf{\Pi}_{\boldsymbol{\theta}}$, and a bad initialization of $\mathbf{\Pi}_{n\times k}$ could lead to the features from different true clusters being compressed. On the other hand, the proposed doubly stochastic membership $\mathbf{\Pi}_{\boldsymbol{\theta}}$ can be initialized deterministically using structures from self-supervised initialized features $\boldsymbol{Z}_{\boldsymbol{\theta}}$ (§2.3). Interestingly, optimizing (4) allows an interpretation of linearizing each point with its neighbors. Empirically as seen in (Table 2), the proposed MLC yields a higher clustering accuracy than NMCE.

2.3 ALGORITHMS

As mentioned, in the MLC objective (4), we parameterize both the representation $\boldsymbol{Z}_{\boldsymbol{\theta}}$ and doubly stochastic membership $\mathbf{\Pi}_{\boldsymbol{\theta}}$ via a neural network. Below we elaborate on how this is done. We summarize the network architecture in Figure 2, and the meta algorithm in Algorithm 1.

Parameterizing $\boldsymbol{Z}_{\boldsymbol{\theta}}$. We follow Yu et al. (2020) and take some existing network architecture as the backbone. We append a few affine layers with non-linearity as the representation head to further transform the output in $\mathbb{R}^d$, followed by a projection layer to respect the unit sphere $\mathbb{S}^{d-1}$ constraint.

Parameterizing $\mathbf{\Pi}_{\boldsymbol{\theta}}$. In subspace clustering, the membership $\mathbf{\Pi}$ given data $\boldsymbol{X}$ often takes the form of $g(\boldsymbol{X})^\top g(\boldsymbol{X})$ for some (linear) transformation g, such as in the inner product kernel (Heckel &

[4]Other choices of regularization are also possible: Essentially any function which achieves its minimum over Ω at the uniform matrix could potentially be used, e.g., the negative entropy function $\sum_{ij}\Pi_{ij}\log(\Pi_{ij})$.

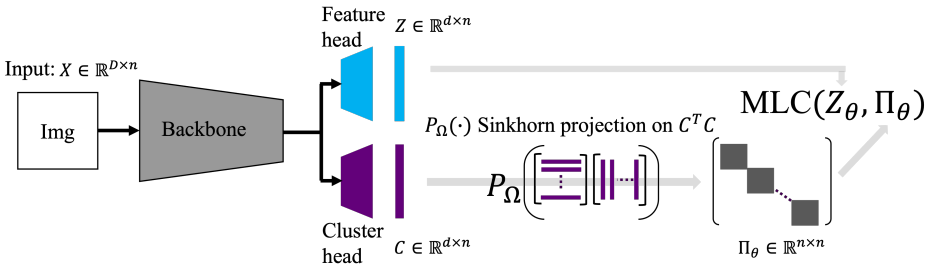


Figure 2: Overall architecture for optimizing the proposed manifold linearizng and clustering (MLC) objective (4). Given n input samples $\boldsymbol{X}$ each lying in $\mathbb{R}^D$, their d-dimensional representation is given by $\boldsymbol{Z}_{\boldsymbol{\theta}}(\boldsymbol{X})$, where $\boldsymbol{\theta}$ denotes network parameters. Further, their doubly stochastic membership matrix $\boldsymbol{\Pi}_{\boldsymbol{\theta}}(\boldsymbol{X})$ is given by taking an inner product kernel of the output of the cluster head $\boldsymbol{C}_{\boldsymbol{\theta}}(\boldsymbol{X})$ followed by a doubly stochastic projection.

Bölcskei, 2015; Ding et al., 2022) where $g = \boldsymbol{I}$ or the least square regression (Lu et al., 2012) where $g = (\boldsymbol{I} + \lambda \boldsymbol{X}^\top \boldsymbol{X})^{-1/2}$. This motivates us to parameterize $g_{\boldsymbol{\theta}}$ by a neural network, and take $\boldsymbol{C}_{\boldsymbol{\theta}}^\top \boldsymbol{C}_{\boldsymbol{\theta}} \in \mathbb{R}^{n\times n}$ as the membership where $\boldsymbol{C}_{\boldsymbol{\theta}}$ is shorthand for $g_{\boldsymbol{\theta}}(\boldsymbol{X})$. Nevertheless, such an $n \times n$ matrix is in general not doubly stochastic, i.e., $\boldsymbol{C}_{\boldsymbol{\theta}}^\top \boldsymbol{C}_{\boldsymbol{\theta}} \notin \Omega$. To obtain a doubly stochastic membership, we further apply a Sinkhorn projection layer $P_{\Omega,\eta}(\cdot)$ (Sander et al., 2021; Eisenberger et al., 2022), which gives our final parameterization of the membership as $\boldsymbol{\Pi}_{\boldsymbol{\theta}} = P_{\Omega,\eta}(\boldsymbol{C}_{\boldsymbol{\theta}}^\top \boldsymbol{C}_{\boldsymbol{\theta}}) \in \Omega$.

Initializing $\boldsymbol{Z}_{\boldsymbol{\theta}}$: Self-supervised Representation Learning via MCR^2. Since the proposed MCR^2 clustering objective (4) is non-convex, it is important to properly initialize both $\boldsymbol{Z}$ and $\boldsymbol{\Pi}$ to converge to good (local) minimum. On the other hand, randomly initialized features are typically far from being ideal, since they may not satisfy the idealized properties (§1), and further may not respect the invariance to augmentation, i.e., the augmented samples should have their representation close to each other. Thus, we adopt the self-supervised training strategy Li et al. (2022)

$$\begin{aligned} \max_{\boldsymbol{\theta}} \quad & \log\det\left(\boldsymbol{I} + \frac{d}{n\epsilon^2}\frac{\boldsymbol{Z}_{\boldsymbol{\theta}} + \boldsymbol{Z}'_{\boldsymbol{\theta}}}{2}\frac{\boldsymbol{Z}_{\boldsymbol{\theta}} + \boldsymbol{Z}'_{\boldsymbol{\theta}}}{2}^\top\right) + \lambda \sum_{i=1}^{n} |\boldsymbol{z}_i^\top \boldsymbol{z}'_i|, \\ \text{s.t.} \quad & \boldsymbol{z}'_i, \boldsymbol{z}_i \in \mathbb{S}^{d-1}, \quad \forall i \in [n], \end{aligned} \tag{5}$$

where for every i, $\boldsymbol{z}_i$ and $\boldsymbol{z}'_i$ are features of different augmentations of the i-th sample. This essentially requires that features from different augmentations of the same sample should be as close as possible, whereas features from different samples should be as uncorrelated as possible.

Initializing $\boldsymbol{\Pi}_{\boldsymbol{\theta}}$. An ideal initialization of $\boldsymbol{\Pi}_{\boldsymbol{\theta}}$ would be such that if $(\boldsymbol{\Pi}_{\boldsymbol{\theta}})_{ij}$ has a high value then points i, j are likely to be from the same true cluster and vice versa. On the other hand, after the self-supervised feature initialization mentioned above, $\boldsymbol{Z}_{\boldsymbol{\theta}}$ already have some structures which we can utilize. Thus, we propose to initialize $\boldsymbol{\Pi}_{\boldsymbol{\theta}}$ with $P_{\Omega,\eta}(\boldsymbol{Z}_{\boldsymbol{\theta}}^\top \boldsymbol{Z}_{\boldsymbol{\theta}})$, which is easily implemented by copying the parameters from $\boldsymbol{Z}_{\boldsymbol{\theta}}$ to $\boldsymbol{C}_{\boldsymbol{\theta}}$ once after the self-supervised initialization of the former, i.e., from the feature head to the cluster head in Figure 2.

Data Augmentation. Beyond initializing $\boldsymbol{Z}_{\boldsymbol{\theta}}$, it is often desirable to incorporate augmentation in optimizing (4). Specifically, from $\{\boldsymbol{X}^{(a)} \in \mathbb{R}^{D\times n}\}_{a=1}^{A}$ the dataset $\boldsymbol{X}$ under A different augmentations, one computes $(\boldsymbol{Z}_{\boldsymbol{\theta}}^{(a)} \in \mathbb{R}^{d\times n}, \boldsymbol{\Pi}_{\boldsymbol{\theta}}^{(a)} \in \mathbb{R}^{d\times n})$ for each augmentation a, and use in (4)

$$\boldsymbol{Z}_{\boldsymbol{\theta}} = P_{\mathbb{S}^{d-1}}\left(\frac{1}{A}\sum_{a=1}^{A} \boldsymbol{Z}_{\boldsymbol{\theta}}^{(a)}\right) \in \mathbb{S}^{d-1}, \quad \boldsymbol{\Pi}_{\boldsymbol{\theta}} = \frac{1}{A}\sum_{a=1}^{A} \boldsymbol{\Pi}_{\boldsymbol{\theta}}^{(a)} \in \Omega. \tag{6}$$

Note that one can benefit from parallelization by putting $\boldsymbol{X}^{(a)}, \boldsymbol{Z}_{\boldsymbol{\theta}}^{(a)}, \boldsymbol{\Pi}_{\boldsymbol{\theta}}^{(a)}$ for each augmentation a on one computing device, since $\boldsymbol{\Pi}_{\boldsymbol{\theta}}^{(a)}$ only depends on $\boldsymbol{X}^{(a)}$ but not from other augmentations.

3 Experiments on Real Datasets

Metrics. To evaluate the clustering quality, we run spectral clustering on learned membership matrix $\boldsymbol{\Pi}$, and report the normalized mutual information (NMI, Strehl & Ghosh (2002)) and

Algorithm 1 MLC: Unsupervised Manifold Linearizing and Clustering

Input: $\boldsymbol{X} \in \mathbb{R}^{D\times n}, \quad \epsilon, \gamma, \eta, \lambda > 0, \quad d, k, n_b, T, A \in \mathbb{Z}_{\geq 0}$

1: initialize $\boldsymbol{Z}_\theta$ by self-supervised representation learning via MCR^2 ▷ (5)
2: initialize $\boldsymbol{\Pi}_\theta$
3: **for** $t = 1, \ldots, T$ **do**
4: $\quad \bar{\boldsymbol{X}} \in \mathbb{R}^{D\times n_b} \leftarrow$ sample a batch from $\boldsymbol{X}$
5: $\quad \bar{\boldsymbol{X}}^{(1)}, \ldots, \bar{\boldsymbol{X}}^{(A)} \leftarrow$ apply A augmentations to $\bar{\boldsymbol{X}}$
6: $\quad \bar{\boldsymbol{Z}}_\theta, \bar{\boldsymbol{\Pi}}_\theta \leftarrow$ forward pass with $\{\bar{\boldsymbol{X}}^{(a)}\}_{a=1}^A$ and network parameters $\boldsymbol{\theta}$ ▷ (6)
7: $\quad \nabla_{\boldsymbol{\theta}}(4) \leftarrow$ backward pass with respect objective (4)
8: $\quad \boldsymbol{\theta} \leftarrow$update $\boldsymbol{\theta}$ using some optimizer on $\nabla_{\boldsymbol{\theta}}(4)$
9: **end for**
10: run spectral clustering on $\boldsymbol{\Pi}_\theta$ to estimate labels $\hat{\boldsymbol{y}}$ of samples

Output: $\boldsymbol{Z}_\theta, \hat{\boldsymbol{y}}$

clustering accuracy (ACC, Lee et al. (2015)), as are commonly used in clustering tasks. To evaluate the learned representation, we define the following metric: for a collection of points $\boldsymbol{W} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_l] \in \mathbb{R}^{d\times l}$ $(l > d)$ with associated singular values $\{\sigma_i\}_{i=1}^d$, define the numerical rank of $\boldsymbol{W}$ as $\arg\min_r \left\{ r : \sum_{i=1}^r \sigma_i^2 / \sum_{i=1}^d \sigma_i^2 > 0.95 \right\}$. Now, one can measure the numerical rank of the learned representaion $\boldsymbol{Z}$, as well as that of each ground-truth cluster[5] of $\boldsymbol{Z}$. A low numerical rank of $\boldsymbol{W}$ implies that points in $\boldsymbol{W}$ lie close to a low-dimensional subspace. We further report the cosine similarity of learned representation, which is simply $|\boldsymbol{z}_i^\top \boldsymbol{z}_j|$ for points i and j, since $\|\boldsymbol{z}_i\| = 1$ by construction in (4). Finally, to compare the efficiency of methods we report the training time in §3.2, where the experiments are run on 2 Nvidia RTX3090 GPUs.

3.1 Comparison with Subspace Clustering

To demonstrate the ability of MLC to cluster the samples and linearize the manifolds, we conduct experiments on CIFAR10, which consists of RGB images from 10 classes such as planes, birds, and deers. As mentioned in §1 subspace clustering methods rely crucially on the assumption that data lie close to a union of linear subspaces, which many real-world dataset may not satisfy. To show that this is the case, we additionally compare the proposed method with subspace clustering methods. As we shall see, applying subspace clustering directly on self-supervised features of CIFAR10 will yield low clustering accuracy. In contrast, MLC is able to *achieve high clustering accuracy*, and moreover, *produce a union-of-orthogonal-subspace representation* on which subspace clustering methods can also achieve high accuracy.

Data. We use the training split of CIFAR10 containing 50000 RGB images, each of size $3 \times 32 \times 32$. We use the augmentation in Appendix C.1 to perform self-supervised representation learning (5) and get $\boldsymbol{Z}_{\text{self}}$. For a fair comparison, the so-learned $\boldsymbol{Z}_{\text{self}}$ are used both as initialization for MLC (line 1 of Algorithm 1), and as the input for subspace clustering methods[6]. In MLC, for each image in each batch we randomly sample $A = 2$ augmentations to apply on the image. As an additional comparison, we also run subspace clustering methods on the features $\boldsymbol{Z}_{\text{MLC}}$ learned by MLC.

Methods. We compare with the elastic-net subspace clustering with active-set solver (EnSC, You et al. (2016a)) and sparse subspace clustering with orthogonal matching pursuit solver (SSC-OMP, You et al. (2016b)), using off-the-shelf implementation provided by the authors[7]. We search the parameters of EnSC over $(\gamma, \tau) \in \{1, 5, 10, 50, 100\} \times \{0.9, 0.95, 1\}$ and those of SSC over $(k_{\max}, \epsilon) \in \{3, 5, 10, 20\} \times \{10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}\}$, and report the run with the highest clustering accuracy for each method. We summarize detailed parameters for MLC in the appendix.

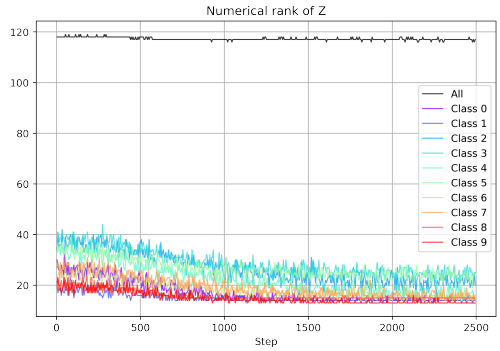
Results. Figure 3 reports the coding rates (as loss terms in (4) and numerical ranks of features learned by MLC as epoch varies. As a first note, the coding rate R of all features (the blue curve in

[5]They are defined by the true labels $\boldsymbol{y}$ (§2), so that the numerical rank metric is decoupled from the quality of learned membership $\boldsymbol{\Pi}$.

[6]The self-supervised features $\boldsymbol{Z}_{\text{self}}$ empirically exhibit some union-of-subspace structure, and are typically used for subspace clustering, as also seen in (Yu et al., 2020, §3.2), (Zhang et al., 2021, §4.2).

[7]`https://github.com/ChongYou/subspace-clustering`

(a) Coding rate of all features R, that of clustered features R_c, and the rate reduction $\Delta R = R - R_c$.



(b) Numerical ranks of all features $\boldsymbol{Z}_\theta$ and features from each ground-truth cluster i, $\{\boldsymbol{z}_j : y(j) = i\}$.

Figure 3: Coding rates (as loss terms in (4)) and numerical ranks (§3.1) of the features learned by MLC on CIFAR10 as epoch varies.

Figure 3a) decreases only slightly as epoch goes, indicating that the overall representation is diverse in the feature space. Indeed, the numerical rank of all features (the dark curve in Figure 3b) stays 118 which is close to the dimension 128 of the feature space. This is in sharp contrast to the deep subspace clustering methods where all the features collapse to a one-dimensional subspace (Haeffele et al., 2020). Moreover, as the coding rate R_c of clustered features (the orange curve in Figure 3a) goes down, the numerical ranks of features from each ground-truth cluster decrease. For instance, the representation from true cluster 3 has a numerical rank of 37 in the first step and 24 in the last step. This implies that most representation gets linearized better and clustered more accurately, even though the MLC objective (4) is unsupervised, i.e., it does not use ground-truth labels $\boldsymbol{y}$. Last but not the least, note that the features within each ground-truth cluster spread well in a low-dimensional subspace, e.g., the numerical ranks for the true clusters at the last step are within $[13, 23]$. This achieves the desired within-cluster diverse property (§1), as opposed to the neural collapse phenomenon that appears with the cross-entropy loss.

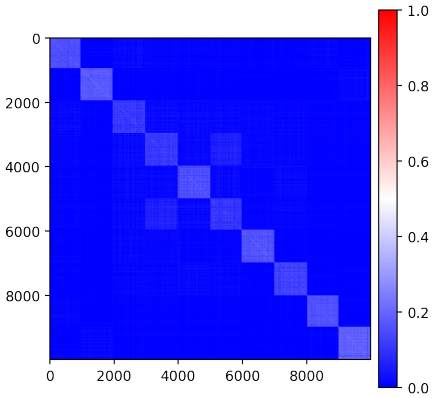


Figure 4: Cosine similarity $|\boldsymbol{Z}_{\text{MLC}}^\top \boldsymbol{Z}_{\text{MLC}}|$ of the features $\boldsymbol{Z}_{\text{MLC}}$ learned by MLC.

Method	Input Data	ACC	NMI
EnSC	$\boldsymbol{Z}_{\text{self}}$	72.2	67.9
	$\boldsymbol{Z}_{\text{MLC}}$	81.5	**79.2**
SSC-OMP	$\boldsymbol{Z}_{\text{self}}$	67.8	64.5
	$\boldsymbol{Z}_{\text{MLC}}$	78.4	76.3
MLC	$\boldsymbol{X}$	**86.3**	78.3

Table 1: Clustering accuracy and normalized mutual information for subspace clustering (EnSC, SSC-OMP) on self-supervised features $\boldsymbol{Z}_{\text{self}}$, features $\boldsymbol{Z}_{\text{MLC}}$ learned by MLC, and manifold clustering (MLC) on $\boldsymbol{X}$, where $\boldsymbol{X}$ is $6 \cdot 10^4$ images from 10 classes of CIFAR10.

To compare MLC with subspace clustering methods, we report clustering accuracy and normalized mutual information for EnSC, SSC-OMP on self-supervised features $\boldsymbol{Z}_{\text{self}}$, features $\boldsymbol{Z}_{\text{MLC}}$ learned by MLC, and MLC on $\boldsymbol{X}$, where $\boldsymbol{X}$ is $6 \cdot 10^4$ images from 10 classes of CIFAR10. In addition we plot the cosine similarity of the features learned by MLC in Figure 4. Remarkably, the highest clustering accuracy is 86.3% achieved by MLC on $\boldsymbol{X}$, which surpasses EnSC (72.2%) and SSC-OMP (67.8%) on $\boldsymbol{Z}_{\text{self}}$ by a large margin, even though $\boldsymbol{Z}_{\text{self}}$ is used both as initialization for MLC and input for EnSC and SSC-OMP. Interestingly, using instead the features $\boldsymbol{Z}_{\text{MLC}}$ learned by MLC, the clustering performance of EnSC and SSC-OMP increases and even becomes comparable to MLC, e.g., EnSC achieves 79.2% normalized mutual information compared to 78.3% of MLC. This suggests that $\boldsymbol{Z}_{\text{MLC}}$ has a union-of-subspace structure that can be utilized by subspace clustering.

Table 2: Clustering accuracy and normalized mutual information on large scale datasets. For a fair comparison, all methods use ResNet-18 as backbone.

Method / Dataset	CIFAR10		CIFAR100-20		CIFAR100-100		Tiny ImageNet-200	
Metrics	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI
DeepCluster $_{(\text{ECCV}'18)}$	37.4	-	18.9	-	-	-	-	-
IIC $_{(\text{ICCV}'19)}$	61.7	51.1	25.7	22.5	-	-	-	-
SCAN $_{(\text{ECCV}'20)}$	87.6	78.7	46.8	45.9	34.3	55.7	-	-
RUC+SCAN $_{(\text{CVPR}'21)}$	**90.3**	-	**53.3**	-	-	-	-	-
IMC-SWAV $_{(\text{Arxiv}'21)}$	89.1	**81.1**	49.0	50.3	43.9	58.3	28.2	52.6
TCR $_{(\text{Arxiv}'22)}$	83.0	76.1	43.7	48.8	-	-	-	-
MLC	86.3	78.3	52.2	**54.6**	**49.4**	**68.3**	**33.5**	**67.5**

Indeed, as seen in Figure 4, features from different clusters tend to have a small similarity, i.e., being orthogonal to each other. This demonstrates the between-cluster discrimination (§1) as desired.

3.2 Comparison with Deep Clustering Methods

We further compare the proposed MLC with state-of-the-art deep clustering methods. Note that most methods reported (all except TCR which is discussed in §2.2) do not aim to learn a union-of-orthogonal-subspace representation, in contrast to MLC. As we will see, MLC achieves clustering accuracy comparable to state-of-the-art methods on large scale datasets with faster computational time, and further surpasses them on datasets of imbalanced clusters.

Compared Methods. We conduct experiments with MLC, SCAN (Van Gansbeke et al., 2020), and IMC-SWAV (Ntelemis et al., 2021).[8] Training details can be found in Appendix C.2. In addition we include the numbers reported from DeepCluster (Caron et al., 2018), IIC (Ji et al., 2018), RUC (Park et al., 2021) and TCR (Li et al., 2022). For a fair comparison, all methods reported use ResNet-18 as the backbone, which is also commonly adopted by other methods.

Datasets. Beyond CIFAR10 (§3.1), we further use CIFAR100-20, CIFAR100-100 and Tiny Imagenet-200 to evaluate the performance of our method. Both CIFAR100-100 and CIFAR100-20 contain the same 50000 train images and 10000 test images with size $32 \times 32 \times 3$, while the former are split into 100 clusters and the latter 20 super clusters. Finally, Tiny ImageNet contains 100000 train images and 10000 test images with size $64 \times 64 \times 3$ split into 200 clusters.

Results on Large-scale Datasets. We report clustering accuracy and normalized mutual information on CIFAR10, CIFAR100-20, CIFAR100-100, and TinyImageNet in Table 2, and we further report running time in minutes for CIFAR100-100 in Table 3. As seen, the highest clustering performance on CIFAR10 is achieved by RUC+SCAN (90.3% ACC) and IMC-SWAV (81.1% NMI), where MLC yields a slightly lower ACC of 86.3% and NMI of 78.3%. We note interesting semantic interpretation for the clustering obtained by MLC in Appendix A. On the other hand, MLC performs comparably with other methods on CIFAR100-20 by achieving an ACC of 52.2% and NMI of 54.6%. Notably, MLC outperforms SCAN and IMC-SWAV on CIFAR100-100 and TinyImageNet-200 by a large margin, while using lower running time: E.g., on CIFAR100-100, MLC yields an accuracy of 49.4% in 291 minutes, whereas IMC-SWAV has 43.9% using 529 minutes, and SCAN has 34.3% in 396 minutes.

Imbalanced Clusters. Note that for CIFAR10 or CIFAR100 each cluster contains approximately the same number of samples. On the other hand, natural images are typically imbalanced, i.e., the clusters have unequal number of samples. To mimic this setting, we take a naive approach to construct the following imbalanced datasets. For the 10 clusters of CIFAR10, we remove half of the samples from odd-numbered clusters (i.e., clusters $1, 3, \ldots, 9$) from both the training and test split. We refer to the reduced dataset Imb-CIFAR10. Likewise we construct Imb-CIFAR100-100. We run

[8] The authors are aware of a preprint (Niu et al., 2021) which approaches image clustering via a combination of self/semi-supervised learning and pseudo-labeling. However, to the best of our effort we are unable to reproduce the numbers reported in this paper using the implementation provided by the authors. We discuss the details in Appendix C.2 and thus do not report their numbers here.

two state-of-the-art methods IMC-SWAV and SCAN as well as the proposed MLC on Imb-CIFAR10 and Imb-CIFAR100-100.

Table 4 shows clustering accuracy on the imbalanced datasets Imb-CIFAR10 and Imb-CIFAR100-100. As a first observation, the clustering accuracy of all methods is lower on the imbalanced datasets than on the balanced counterparts, as expected. Notably, MLC suffers from the least performance drop, e.g., when moving from CIFAR10 to Imb-CIFAR10 the accuracy of MLC drops from 86% to 80%, whereas that of SCAN and IMC-SWAV decreases from above 87% to below 66%.

Table 3: Running time in minutes and clustering accuracy on CIFAR100-100. For a fair comparison, all methods use ResNet-18 as backbone.

Method / Metric	Running Time				ACC
Stage	I	II	III	Total	
SCAN $_{(\text{ECCV}'20)}$	308.3	33.3	54.7	396.3	34.3
IMC-SWAV $_{(\text{Arxiv}'22)}$	529.4	-	-	529.4	43.9
MLC	266.7	17.7	-	284.4	48.3

Table 4: Clustering accuracy on imbalanced datasets: (a) Imb-CIFAR10, (b) Imb-CIFAR100-100. For a fair comparison, all methods use ResNet-18 as backbone.

Method / Dataset	(a)	(b)
IMC-SWAV $_{(\text{Arxiv}'21)}$	65.7	38.2
SCAN $_{(\text{ECCV}'20)}$	62.9	31.1
MLC	**80.0**	**46.1**

4 Conclusion

This paper studies the problem of simultaneously clustering and learning an union-of-orthogonal-subspace representation for data, when data lies close to a union of low-dimensional manifolds. To address the problem we propose an objective based on *maximal coding rate reduction* and *doubly stochastic* membership inspired by the state-of-the-art subspace clustering results. We provide an efficient and effective parameterization of the membership variables as well as a meta-algorithm to optimize the representation and membership jointly. We further conduct experiments on datasets with larger number of clusters and imbalanced clusters and show that the proposed method achieves state-of-the-art performance. We believe that our work provides a general and unified framework for unsupervised learning of structured representations for multi-modal data.

Ethics Statement

All authors agree and will adhere to the conference's Code of Ethics. We do not anticipate any potential ethics issues regarding the research conducted in this work.

Reproducibility Statement

Settings and implementation details of network architectures, optimization methods, and some common hyper-parameters are described in the Appendix C. We will also make our source code available upon request by the reviewers or the area chairs.

References

Mahdi Abavisani and Vishal M Patel. Deep multimodal subspace clustering networks. *IEEE J. Sel. Top. Signal Process.*, 12(6):1601–1614, April 2018.

David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, June 2006.

Bahman Bahmani, Benjamin Moseley, Andrea Vattani, Ravi Kumar, and Sergei Vassilvitskii. Scalable K-Means++. *Proceedings VLDB Endowment*, 5(7), March 2012.

Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-Invariance-Covariance regularization for Self-Supervised learning. In *International Conference on Learning Representations*, 2022.

P S Bradley, O L Mangasarian, and W N Street. Clustering via concave minimization. In *Advances in neural information processing systems*, 1996.

Joan Bruna and Stéphane Mallat. Invariant scattering convolution networks. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1872–1886, August 2013.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *European conference on computer vision*, pp. 132–149, July 2018.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Adv. Neural Inf. Process. Syst.*, 33: 22243–22255, 2020.

Tianjiao Ding, Derek Lim, Rene Vidal, and Benjamin D Haeffele. Understanding doubly stochastic clustering. In *the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 5153–5165. PMLR, 2022.

Marvin Eisenberger, Aysim Toker, Laura Leal-Taixé, Florian Bernard, and Daniel Cremers. A unified framework for implicit sinkhorn differentiation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 509–518, 2022.

Ehsan Elhamifar and Rene Vidal. Sparse subspace clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2790–2797, June 2009.

Ehsan Elhamifar and René Vidal. Sparse manifold clustering and embedding. *Adv. Neural Inf. Process. Syst.*, 24, 2011.

Ehsan Elhamifar and Rene Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(11):2765–2781, 2013.

Forgey. Cluster analysis of multivariate data: Efficiency vs. interpretability of classification. *Biometrics*, 1965.

Benjamin D Haeffele, Chong You, and René Vidal. A critique of Self-Expressive deep subspace clustering. In *International Conference on Learning Representations*, 2020.

Reinhard Heckel and Helmut Bölcskei. Robust subspace clustering via thresholding. *IEEE Trans. Inf. Theory*, 61(11):6320–6342, 2015.

R C Jancey. Multidimensional group analysis. *Australian Journal of Botany*, 14:127–130, 1966.

Pan Ji, Tong Zhang, Hongdong Li, Mathieu Salzmann, and Ian Reid. Deep subspace clustering networks. In *Advances in Neural Information Processing Systems*, volume 2017-Decem, pp. 24–33, 2017.

Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *IEEE/CVF International Conference on Computer Vision*, July 2018.

Mohsen Kheirandishfard, Fariba Zohrizadeh, and Farhad Kamangar. Deep low-rank subspace clustering. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, June 2020.

Minsik Lee, Jieun Lee, Hyeogjin Lee, and Nojun Kwak. Membership representation for detecting block-diagonal structure in low-rank or sparse subspace clustering. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 07-12-June:1648–1656, 2015.

Chun-Guang Li, Chong You, and René Vidal. On geometric analysis of affine sparse subspace clustering. *IEEE J. Sel. Top. Signal Process.*, 12(6):1520–1533, December 2018.

Zengyi Li, Yubei Chen, Yann LeCun, and Friedrich T Sommer. Neural manifold clustering and embedding. January 2022.

Derek Lim, René Vidal, and Benjamin D Haeffele. Doubly stochastic subspace clustering. November 2020.

Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. Robust recovery of subspace structures by low-rank representation. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):171–184, January 2013.

S Lloyd. Least squares quantization in PCM. Technical report, Bell Laboratories, 1957.

Can-Yi Lu, Hai Min, Zhong-Qiu Zhao, Lin Zhu, De-Shuang Huang, and Shuicheng Yan. Robust and efficient subspace segmentation via least squares regression. In *European conference on computer vision*, pp. 347–360. Springer, 2012.

Yi Ma, Harm Derksen, Wei Hong, and John Wright. Segmentation of multivariate mixed data via lossy data coding and compression. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(9):1546–1562, 2007.

J MacQueen. Some methods for classification and analysis of multivariate observations. In *5th Berkeley Symp. Math. Statist. Probability*, pp. 281–297, 1967.

Chuang Niu, Hongming Shan, and Ge Wang. SPICE: Semantic pseudo-labeling for image ClustEring. March 2021.

Foivos Ntelemis, Yaochu Jin, and Spencer A Thomas. Information maximization clustering via Multi-View Self-Labelling. March 2021.

Vardan Papyan, X Y Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proc. Natl. Acad. Sci. U. S. A.*, 117(40):24652–24663, October 2020.

Sungwon Park, Sungwon Han, Sundong Kim, Danu Kim, Sungkyu Park, Seunghoon Hong, and Meeyoung Cha. Improving unsupervised image clustering with robust learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12278–12287, 2021.

Vishal M Patel and Rene Vidal. Kernel sparse subspace clustering. In *2014 IEEE International Conference on Image Processing (ICIP)*. IEEE, October 2014.

Xi Peng, Jiashi Feng, Shijie Xiao, Jiwen Lu, Zhang Yi, and Shuicheng Yan. Deep sparse subspace clustering. September 2017.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *Ann. Math. Stat.*, 22(3): 400–407, 1951.

Daniel P Robinson, Rene Vidal, and Chong You. Basis pursuit and orthogonal matching pursuit for subspace-preserving recovery: Theoretical analysis. December 2019.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient content-based sparse attention with routing transformers. *Trans. Assoc. Comput. Linguist.*, 9:53–68, February 2021.

Michael E Sander, Pierre Ablin, Mathieu Blondel, and Gabriel Peyré. Sinkformers: Transformers with doubly stochastic attention. October 2021.

Mahdi Soltanolkotabi and Emmanuel J Candés. A geometric analysis of subspace clustering with outliers. *Annals of statistics*, 40(4):2195–2238, 2012.

Mahdi Soltanolkotabi, Ehsan Elhamifar, and Emmanuel J Candes. Robust subspace clustering. *Ann. Stat.*, 42(2):669–699, 2014.

Strehl and Ghosh. Cluster ensembles – a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, 2002.

Tom Tirer and Joan Bruna. Extended unconstrained features model for exploring deep neural collapse. In *International Conference on Machine Learning*, February 2022.

Manolis Tsakiris and Rene Vidal. Theoretical analysis of sparse subspace clustering with missing entries. In Jennifer Dy and Andreas Krause (eds.), *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4975–4984. PMLR, 2018.

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*. Springer, 2020.

Yining Wang, Yu-Xiang Wang, and Aarti Singh. A deterministic analysis of noisy sparse subspace clustering for dimensionality-reduced data. In *International Conference on Machine Learning*, 2015.

Yu-Xiang Wang and Huan Xu. Noisy sparse subspace clustering. *Journal of Machine Learning Research*, 2016.

Chong You, Chun Guang Li, Daniel P Robinson, and Rene Vidal. Oracle based active set algorithm for scalable elastic net subspace clustering. pp. 3928–3937, 2016a.

Chong You, Daniel P Robinson, and René Vidal. Scalable sparse subspace clustering by orthogonal matching pursuit. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2016b.

Chong You, Chun-Guang Li, Daniel Robinson, and Rene Vidal. Is an affine constraint needed for affine subspace clustering? In *International Conference on Computer Vision (ICCV)*. IEEE, October 2019.

Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. August 2017.

Yaodong Yu, Kwan Ho Ryan Chan, Chong You, Chaobing Song, and Yi Ma. Learning diverse and discriminative representations via the principle of maximal coding rate reduction. In *Neural Information Processing Systems*, June 2020.

Junjian Zhang, Chun-Guang Li, Chong You, Xianbiao Qi, Honggang Zhang, Jun Guo, and Zhouchen Lin. Self-Supervised convolutional subspace clustering network. In *IEEE/CVF conference on computer vision and pattern recognition*, 2019.

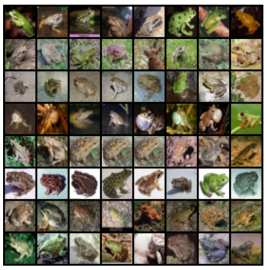
Shangzhi Zhang, Chong You, René Vidal, and Chun-Guang Li. Learning a self-expressive network for subspace clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, October 2021.

Jinxin Zhou, Xiao Li, Tianyu Ding, Chong You, Qing Qu, and Zhihui Zhu. On the optimization landscape of neural collapse under MSE loss: Global optimality with unconstrained features. In *International Conference in Machine Learning*, March 2022.

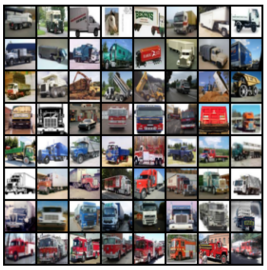
A SEMANTIC INTERPRETABILITY OF THE LEARNED REPRESENTATION AND CLUSTERS ON CIFAR10

Recall that MLC is designed to perform clustering while learning a union-of-orthogonal-subspace representation (§1), where each cluster defines a low-dimensional subspace. Therefore, we further visualize the different directions within each learned cluster or subspace. Specifically, after a final clustering is obtained (line 10 of Algorithm 1), we take the features from each learned cluster and apply Principal Component Analysis (PCA) to them to obtain the first 8 principal components. These correspond to the 8 rows for each cluster in Figure 5. Recall that the principal components are mutually orthogonal, indicating uncorrelated directions within one cluster. To visualize those directions or principal components in images, we take the features that are closest to the principal components and visualize the corresponding original images.

Interestingly, the rows of images corresponding to principal components appear to exhibit some semantic 'concepts'. For example in Figure 5b, row 1 and 8 are respectively white and red trucks, while row 4 are the trucks that ship sand or mud; row 1 of Figure 5d are deers with trees as background. This further suggests that the learned embedding seems to preserve distance within each



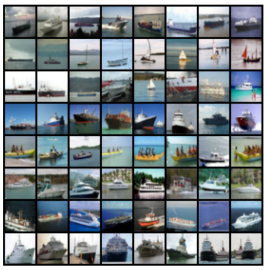
(a) Learned cluster 1



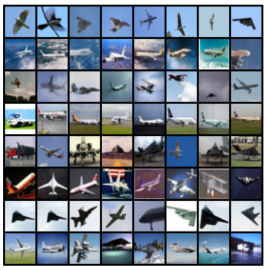
(b) Learned cluster 2

(c) Learned cluster 3

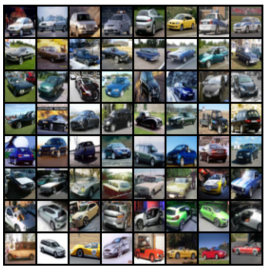
(d) Learned cluster 4



(e) Learned cluster 5



(f) Learned cluster 6



(g) Learned cluster 7

(h) Learned cluster 8

(i) Learned cluster 9

Figure 5: Principal images (defined in §A) of clusters learned by MLC on CIFAR10.

cluster (as desired in §1), i.e., images that are close/far in semantic meaning will be close/far in the feature space. Note however, that some learned clusters do not align fully with the ground-truth labels. For instance, rows 1 and 3 of Figure 5h are cats while all other rows in this cluster are dogs. On the other hand, one may argue that Figure 5h are a cluster of cats and dogs of light colors, whereas Figure 5c is a cluster of those of brown colors, which could be a semantically meaningful clustering even though it does not align with the ground-truth labels.

B SIMULATION ON SYNTHETIC UNION-OF-MANIFOLD DATA

We perform simulations to visualize the properties of the proposed manifold learning and clustering method. As seen in Figure 1a, we generate data $\boldsymbol{X}$ from two manifolds on the sphere $\mathbb{S}^2$, each consisting of 200 samples. The points from the first manifold (green) take the form $\boldsymbol{x}_i = \left[\cos\left(A\sin(\omega\phi_i)\right)\cos\phi_i, \cos\left(A\sin(\omega\phi_i)\right)\sin\phi_i, \sin\left(A\sin(\omega\phi_i)\right)\right]^\top + \boldsymbol{\epsilon}_i$, where $A = 0.2$ and $\omega = 5$ sets the curvature of the manifold, $\boldsymbol{\epsilon}_i \sim \mathcal{N}(\boldsymbol{0}, 0.05\boldsymbol{I}_3)$ is the additive noise, and we take $\phi_i = \frac{2\pi i}{100}$ for $i = 1, \ldots, 100$ to generate 100 points. On the other hand, the points from the second manifold (blue) are simply 100 samples from $\mathcal{N}([0, 0, 1]^\top, 0.05\boldsymbol{I}_3)$. We take the feature dimension $d = 3$ to be equal to he input dimension $D = 3$. We paramterize both the feature head $f_{\boldsymbol{\theta}}$ and the cluster head $g_{\boldsymbol{\theta}}$ to be a simple fully-connected network with 100 hidden neurons, followed by a Rectified Linear Unit as non-linearity and a projection operator onto the sphere $\mathbb{S}^2$. Figures 1b to 1d report the features $\boldsymbol{Z}_{\boldsymbol{\theta}}$ with random initialization (i.e., before line 1 of Algorithm 1), with self-supervised initialization, and at convergence of MLC. Notably, despite $\boldsymbol{Z}_{\boldsymbol{\theta}}$ being noisy and only approximately piece-wise linear, as epoch goes $\boldsymbol{Z}_{\boldsymbol{\theta}}$ gradually transform to two linear subspaces: the green points converge to a 2-dimensional subspace (intersected with $\mathbb{S}^2$) and the blue points converge to a 1-dimension subspace.

C TRAINING DETAILS ON REAL DATASETS

C.1 TRAINING DETAILS OF MLC

As said, we use ResNet-18 as the backbone for experiments on CIFAR10, CIFAR100-20, CIFAR100-100 and Tiny-ImageNet-200, and the imbalanced counterparts Imb-CIFAR10, Imb-CIFAR100-100. We also fix the batch size to be 1024 in all experiments. In self-supervised initialization of $\boldsymbol{Z}_{\boldsymbol{\theta}}$ (line 1 of Algorithm 1), we use the precision (§2.1) parameter $\epsilon^2 = 0.2$, a LARS optimizer (You et al., 2017) (as is also done in (Chen et al., 2020; Li et al., 2022)) with a learning rate of 0.3 and trained MLC for 1000 epochs. On the other hand, in the training of MLC objective, we use $\epsilon^2 = 0.1$ and $\gamma = 0.05$, an SGD optimizer (Robbins & Monro, 1951) with a learning rate of 0.01 and momentum of 0.9, and $\eta = 0.175$ for the entropy regularization in the Sinkhorn projection (Eisenberger et al., 2022) layer $P_{\Omega,\eta}(\cdot)$. Finally, for all experiments, we use the augmentation applied in (Bardes et al., 2022) detailed below in PyTorch code.

1. transforms.RandomResizedCrop(32,scale=(0.04, 1.0))
2. transforms.RandomHorizontalFlip(p=0.5)
3. transforms.RandomGrayscale(p=0.2)
4. transforms.RandomApply([transforms.ColorJitter(0.4, 0.4, 0.4, 0.1)], p=0.8)
5. GaussianBlur(p=0.1)

C.2 TRAINING DETAILS OF OTHER METHODS

To compare with state-of-the-art methods, we use the code released by the authors. As mentioned, the preprint (Niu et al., 2021) proposed a method SPICE that appears to achieve state-of-the-art performance in image clustering. We tried to reproduce their results on CIFAR-100-20 using the official implementation[9]. However, the provided implementation ran into a few errors, which are also discussed in [10]. Despite our best effort to fix those issues, the experiments yield 14% accuracy

[9] https://github.com/niuchuangnn/SPICE, commit 5eba538.

[10] https://github.com/niuchuangnn/SPICE/issues/27,https://github.com/niuchuangnn/SPICE/issues/31

on CIFAR100-20 as opposed to the 53% reported in their paper. Therefore, in this work, we does not include SPICE in our comparison.

D SENSITIVITY OF MLC TO RANDOM SEED

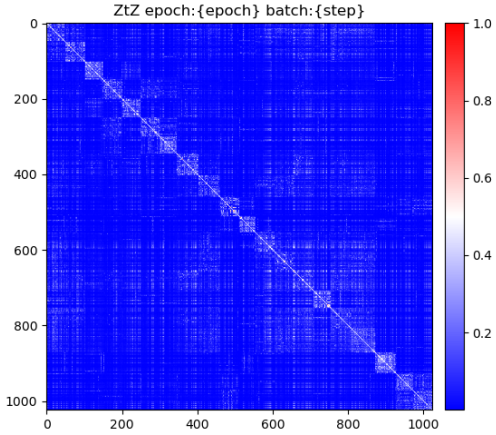
It has been noticed in the previous literature (Van Gansbeke et al., 2020) that random seeds can largely influence the performance of image clustering. In this section, we study the influence random seeds have on our method. We conduct experiment on CIFAR100-100, using standard training details which we described above. We use CIFAR100-100 because it is one of the hardest dataset in our work, so the influence of random seed should be most obvious. We report in Table 5 the accuracy of MLC on CIFAR100-100 with different seeds. We observe that the choice of seed has very little impact on performance.

Random Seed	1	5	10	15	100
Accuracy	0.483	0.480	0.488	0.482	0.484

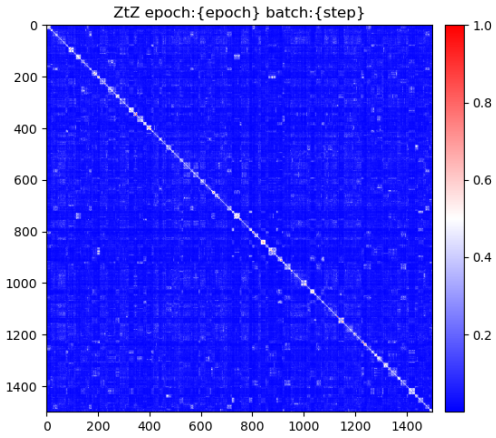
Table 5: Ablation study on varying random seeds.

E MORE RESULTS ON MEMBERSHIP LEARNED BY MANIFOLD CLUSTERING

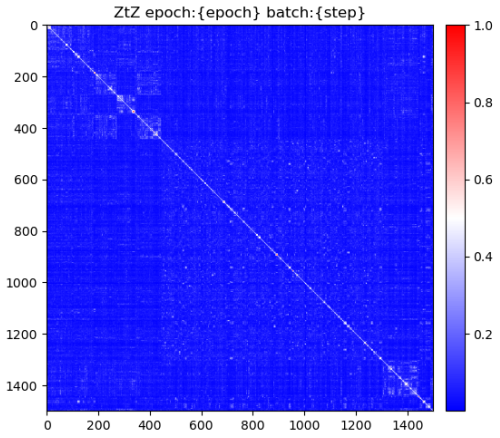
In this section, we present more results on the membership learned by manifold clustering on other datasets like CIFAR100-20, CIFAR100-100 and TinyImageNet-200. We present these results in Fig **??**. The learned representation and clusters form a block diagonal structure, showing that the feature by clustering are orthogonal to each other. This verifies the between-cluster discrimination (§1) as desired on larger scale of data.



(a) CIFAR100-20

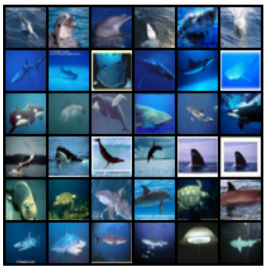


(b) CIFAR100-100



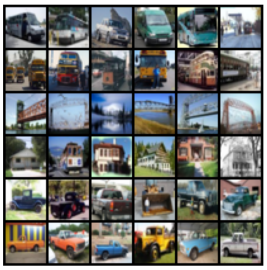
(c) Tiny ImageNet-200

Figure 6: Cosine similarity $|\boldsymbol{Z}_{\mathrm{MLC}}^{\top}\boldsymbol{Z}_{\mathrm{MLC}}|$ of the features $\boldsymbol{Z}_{\mathrm{MLC}}$ learned by MLC on more complicated datasets: CIFAR100-20, CIFAR100-100, Tiny ImageNet-200

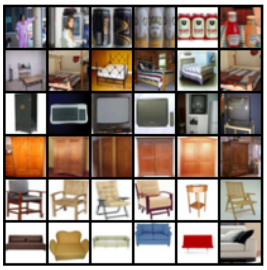


(a) Learned cluster 1

(b) Learned cluster 2

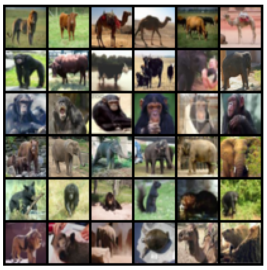


(c) Learned cluster 3



(d) Learned cluster 4

(e) Learned cluster 5



(f) Learned cluster 6

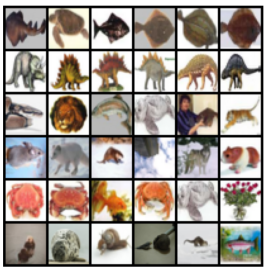
(g) Learned cluster 7

(h) Learned cluster 8

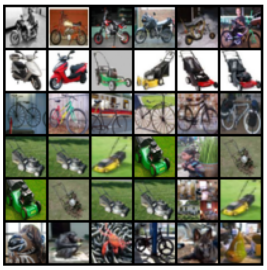
(i) Learned cluster 9

(j) Learned cluster 10

(k) Learned cluster 11

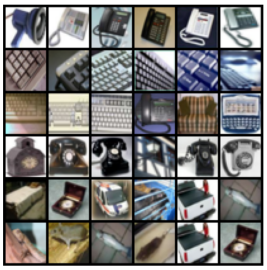


(l) Learned cluster 12

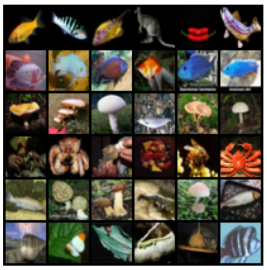


(a) Learned cluster 13

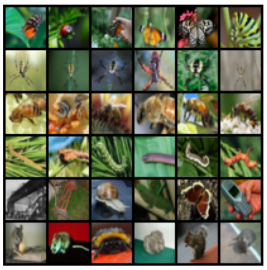
(b) Learned cluster 14



(c) Learned cluster 15

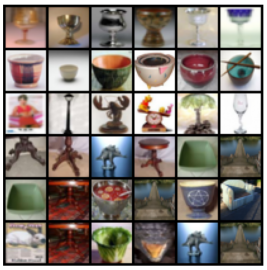


(d) Learned cluster 16

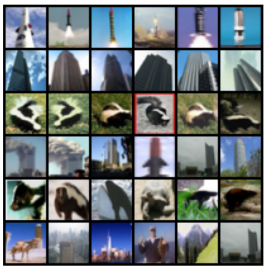


(e) Learned cluster 17

(f) Learned cluster 18



(g) Learned cluster 19



(h) Learned cluster 20

Figure 8: Principal images of clusters learned by MLC on CIFAR100-20.