Unsupervised Manifold Linearizing and Clustering

Tianjiao Ding¹ Shengbang Tong² Kwan Ho Ryan Chan¹ Xili Dai³ Yi Ma² Benjamin D. Haeffele¹

Abstract

We consider the problem of simultaneously clustering and learning a linear representation of data lying close to a union of low-dimensional manifolds, a fundamental task in machine learning and computer vision. When the manifolds are assumed to be linear subspaces, this reduces to the classical problem of subspace clustering, which has been studied extensively over the past two decades. Unfortunately, many real-world datasets such as natural images can not be well approximated by linear subspaces. On the other hand, numerous works have attempted to learn an appropriate transformation of the data, such that data is mapped from a union of general non-linear manifolds to a union of linear subspaces (with points from the same manifold being mapped to the same subspace). However, many existing works have limitations such as assuming knowledge of the membership of samples to clusters, requiring high sampling density, or being shown theoretically to learn trivial representations. In this paper, we propose to optimize the Maximal Coding Rate Reduction metric with respect to both the data representation and a novel doubly stochastic cluster membership, inspired by state-of-the-art subspace clustering results. We give a parameterization of such a representation and membership, allowing efficient mini-batching and one-shot initialization. Experiments on CIFAR-10, -20, -100, and TinyImageNet-200 datasets show that the proposed method is much more accurate and scalable than state-of-the-art deep clustering methods, and further learns a latent linear representation of the data.¹

1. Introduction

1.1. Clustering: from Linear to Non-linear Models

Clustering is a fundamental problem in machine learning, allowing one to group data into clusters based on assumptions about the geometry of each cluster. As early as the 1950s, the classic $k$-means [43, 23, 30, 46] algorithm emerged to cluster data that concentrate around distinct centroids, with numerous variants [8, 2, 4] following. This assumption of distinct centroids was later generalized in subspace clustering methods, which aim to cluster data lying close to a union of low-dimensional linear (or affine) subspaces. This motivated numerous lines of research in the past two decades, leading to various formulations [20, 22, 44, 41, 28, 75, 40, 33] with efficient algorithms [76, 75, 15] and theoretical guarantees on the correctness of the clustering [58, 59, 71, 72, 36, 63, 74, 55, 69]. Subspace clustering has been used in a wide range of applications, such as segmenting image pixels [45, 70, 42], video frames [65, 61, 37], or rigid-body motions [66, §11], along

¹Note, a centroid can be seen as a 0-dimensional affine subspace.
with clustering face images [24, 29, 22] or human actions [73, 25, 50].

However, while subspace clustering methods have achieved state-of-the-art performance for certain tasks and datasets, the geometric assumption upon which they rely (namely that the datapoints lie on a union of linear or affine subspaces) is often grossly violated for common high-dimensional datasets. For instance, even in a dataset as simple as MNIST hand-written digits [34], images of a single digit do not lie close to a low-dimensional subspace; directly applying subspace clustering methods thus fails. Instead, a more natural idea is to assume that each cluster is defined by a non-linear low-dimensional manifold and to learn or design a non-linear transformation of the data so that points from one manifold are mapped to one linear subspace. In some cases, one may be able to hand-craft an appropriate transformation of the data, with polynomial or exponential kernel mappings being examples that have been explored in the literature [21], and the authors of [39] show that a subspace clustering method achieves 99% clustering accuracy on MNIST when the data is passed through the scattering-transform [9].

Unfortunately though, hand-crafted design requires one to assume specific and simple families of manifolds which is often unrealistic and challenging to apply on complicated data such as natural images. On the other hand, the authors of [21] propose to cluster data via treating a local neighborhood of the manifold approximately as a linear subspace and applying subspace clustering techniques to local neighborhoods. However, this method requires sufficient sampling density to succeed, which implies a prohibitive number of samples when the manifolds have moderate dimension or curvature. More recently, numerous works propose to learn an appropriate transformation of the data via deep networks and then perform subspace clustering in a latent feature space [53, 31, 1, 80, 32]. Unfortunately, it has been shown that many of these formulations are ill-posed and provably learn trivial representations, with much of the claimed benefit coming from ad-hoc post-processing rather than the method itself [27]. This motivates the one of the primary questions we consider here:

**Question 1.** Can we efficiently transform data near a union of low-dimensional manifolds, so that the transformed data lie close to a union of low-dimensional linear subspaces to allow for easy clustering?

### 1.2. Learning Diverse and Discriminative Features: from Supervised to Unsupervised Learning

Meanwhile, learning a compact representation from multi-modal data has been a topic of its own interest in machine learning [7]. An ideal property of the learned representation is between-cluster discrimination, namely, features from different clusters should be well separated, which is often pursued via a loss such as the classic cross-entropy (CE) objective. However, an important yet often ignored property is that the learned representation maintains within-cluster diversity. This allows distances of samples within a cluster to be preserved under the learned transformation, which may facilitate downstream tasks such as generation [16], denoising [68], and semantic interpretation [78, §B.3.1] (see also §A). Unfortunately, the representation learned by CE fails to achieve this property and exhibits neural collapse, a phenomenon discovered by [51] with extensive theoretical and empirical analysis [47, 85, 62, 83] (even for non-CE objectives [84]), where latent features from one cluster tend to collapse to a single point. In contrast, [78] recently proposed Maximal Coding Rate Reduction (MCR$^2$) as an objective to pursue both of the mentioned ideal properties. In particular, MCR$^2$ learns a union-of-orthogonal-subspaces representation: features from each cluster spread uniformly in a low-dimensional subspace (compact & within-cluster diverse), and the subspaces corresponding to different clusters are orthogonal to each other (between-cluster discriminative). Nevertheless, MCR$^2$ requires ground-truth labels to learn such a representation. This leads to our second question of interest:

**Question 2.** For data lying close to a union of manifolds, can we learn a union-of-orthogonal-subspaces representation, without access to the ground-truth labels?

### 1.3. Our Contributions

To address the two interrelated questions, we start with the basic idea of blending the philosophies from MCR$^2$ and subspace clustering to explore the best of both worlds. This idea leads us to the following contributions.

- **Formulation (§2):** We propose Manifold Linearizing and Clustering (MLC) objective (4), which optimizes the MCR$^2$ loss over both the representation and a novel doubly stochastic cluster membership. The latter consists of pair-wise similarities between samples, and it is constrained to be doubly stochastic, inspired by state-of-the-art subspace clustering results [39, 18].

- **Algorithm (§3):** We describe how to parameterize and initialize the representation and membership, as well as to optimize MLC (4). Even though the membership is doubly stochastic, which may appear large in size and hard to constrain, we give an efficient parameterization of it that allows for mini-batching and notably one-shot initialization. That is, the membership is initialized with no additional training whatsoever leveraging already-initialized representation, which is stable, structured, and efficient.

---

6In this paper, we use ‘representation’ and ‘feature’ interchangeably to mean the image of the data under a (learned) transformation.
• Experiments (§4): On CIFAR-10, we demonstrate that MLC learns a union-of-orthogonal-subspaces representation, and achieves more accurate clustering than state-of-the-art subspace clustering methods. Moreover, on CIFAR-10, -20, -100, and TinyImageNet-200, we show that MLC yields higher clustering accuracy using less running time than state-of-the-art deep clustering methods, even when there are many or imbalanced clusters.

1.4. Additional Related Work

Beyond the above, we make connections to a few emerging deep-learning-based works related to this paper.

Self-supervised Representation Learning. An important line of research that learns a representation without using ground-truth labels is that of self-supervised learning. It has seen remarkable recent progress thanks to the so-called joint-embedding approach. The basic idea of the latter is to learn a representation such that augmentations of the same input have similar features, while features from different inputs do not collapse to a single point. Extending this idea, self-supervised methods such as SimCLR [12], BYOL [26], and VICReg [5] are able to learn representations on par with those obtained from supervised learning methods; see [57] for an excellent review. Encouraging as it may sound, these methods do not aim to learn a union-of-orthogonal-subspaces representation, nor do they explicitly model clustering in their design. Nevertheless, we shall see that self-supervised methods are key stepping stones for the proposed method, as they will be used to initialize parts of our model.

Clustering and Representation Learning. Numerous works have proposed to jointly perform clustering and representation learning, leveraging the success of self-supervised learning. Roughly speaking, most methods consider the following two steps. The first step is to use self-supervised learning to initialize the representation. Indeed, state-of-the-art methods such as SCAN [64] and SPICE [48] adopt SimCLR [12] and MoCoV2 [14] as their pre-trained features. Starting from the initial representation, the second step is to iteratively refine the representation and clustering, using the idea of pseudo-labeling [10, 64, 52, 48]. Despite the promising clustering performance, the representation learned by these methods is not constrained to be both between-cluster discriminative and within-cluster diverse. In contrast, our approach learns a representation with these two ideal properties (Figure 4) and also achieves state-of-the-art clustering performance (Tables 3, 4 and 6). Finally, the work most closely related to this paper is that of Neural Manifold Clustering and Embedding (NMCE) [38] – we note similarities and differences in terms of formulation, algorithm, and empirical performance at the end of §2.2.

2. Formulation

We begin by making clear the problem of interest.

Problem 1 (Unsupervised Manifold Linearizing and Clustering). Suppose \( X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^{D \times n} \) is a dataset of \( n \) points lying on an union of \( k \) low-dimensional manifolds \( \bigcup_{j=1}^{k} M_j \). Given \( X \), we aim to simultaneously

1. Cluster the samples: find \( \hat{y} \) such that \( x_i \in M_{\hat{y}(i)} \);

2. Learn a linear representation: find a transformation \( f : \mathbb{R}^{D} \rightarrow \mathbb{R}^{d} \), such that features \( f(x_i) \)'s from the same cluster spread uniformly in a low-dimensional linear subspace, and the subspaces arising from different clusters are orthogonal.

In §2.1, we review the principle of Maximal Coding Rate Reduction (MCR\(^2\)) which is designed to learn ideal representations in a supervised manner, i.e., when the ground-truth membership is given. Then in §2.2, we discuss the challenges of simultaneous clustering and learning a representation (i.e., addressing Problem 1 in its entirety), for which we propose our MLC objective (4). Later in §3, we further give an algorithm to optimize MLC (4).

2.1. Supervised Manifold Linearizing via MCR\(^2\)

When the cluster membership is given as supervision, MCR\(^2\) [78] is designed to solve part 2) of Problem 1. To begin with, let \( f_0 : \mathbb{R}^{D} \rightarrow \mathbb{S}^{d-1} \) be a transformation parameterized by a neural network; this in turn gives the features \( Z := [z_1, \ldots, z_n] \in \mathbb{R}^{d \times n} \) of data with \( z_i := f_0(x_i) \in \mathbb{S}^{d-1} \). MCR\(^2\) aims to learn an ideal representation by utilizing the coding rate measures. Define the coding rate

\[
R(Z; \epsilon) := \log \det \left( I + \frac{d}{m \epsilon^2} ZZ^\top \right).
\]

Intuitively\(^7\), \( R(Z; \epsilon) \) measures some volume of features in \( Z \) up to \( \epsilon > 0 \) precision, so maximizing it would diversify features of all the samples. Likewise, one can apply the measure to features of each cluster and take weighted mean over the clusters; namely, define \( R_c(Z, \Pi; \epsilon) \) as

\[
\sum_{j=1}^{k} \frac{1}{n_j} \log \det \left( I + \frac{d}{n_j \epsilon^2} Z \text{Diag}(\Pi_j) Z^\top \right).
\]

\(^7\)For a rigorous treatment, see for example [45, §2.1].
Here $\Pi = [\Pi_1, \ldots, \Pi_k] \in \mathbb{R}^{n \times k}$ is a given point-cluster membership such that $\Pi_{ij} = 1$ if $x_j \in M_i$ and $\Pi_{ij} = 0$ otherwise, $1$ is a vector of all ones so $(\Pi_i, 1)$ is the number of points in cluster $j$, and for any $v \in \mathbb{R}^n$, $\text{Diag}(v)$ denotes a diagonal matrix with the entries of $v$ along the diagonal. Minimizing $R_c(Z, \Pi; \epsilon)$ thus pushes features in each cluster to stay close. $\text{MCR}^2$ learns an ideal representation by maximizing the coding rate reduction (hence the acronym)

$$\max_\theta R(Z; \epsilon) - R_c(Z, \Pi; \epsilon) \quad \text{s.t.} \quad Z = f_\theta(X) \quad (1)$$

Notably, it has been shown that given $\Pi$, the features obtained by maximizing the rate reduction has the property that the features of each cluster spread uniformly within a subspace (within-cluster diverse), and subspaces from different clusters are orthogonal (between-cluster discriminative), under relatively mild assumptions [78, Theorem 2.1].

2.2. Unsupervised Manifold Linearizing and Clustering

While $\text{MCR}^2$ is designed to learn ideal representations ($\S$1) when the membership $\Pi$ is given, here we are interested in the unsupervised setting where one does not have access to membership annotations. To address both parts 1) and 2) of Problem 1, a natural idea is to optimize $\text{MCR}^2$ over both the representation $Z$ and membership $\Pi$ via

$$\max_{\theta, \Pi \in \Pi_o} R(Z; \epsilon) - R_c(Z, \Pi; \epsilon) \quad \text{s.t.} \quad Z = f_\theta(X). \quad (2)$$

Here $\Pi_o := \{ \Pi \in \mathbb{R}^{n \times k} : \forall i \in [n], \exists \hat{y}(i) \text{ s.t. } \Pi_{i\hat{y}(i)} = 1 \text{ and } \Pi_{ij} = 0 \text{ for } j \neq \hat{y}(i) \}$ is the set of all “hard” assignments, i.e., each row of $\Pi$ 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 $\Pi$. Further, a second challenge is the chicken-and-egg nature of this problem: If one already has an ideal representation $Z$, then existing subspace clustering methods can be applied on $Z$ to estimate the membership. Likewise, if one is given a smooth membership $\Pi$ of clusters, then solving (1) would lead to an ideal representation. However, the $Z$ and $\Pi$ at the beginning of optimization are typically far from ideal.

In the rest of this section, we propose a so-called doubly stochastic membership to deal with the combinatorial challenge. To tackle the chicken-and-egg nature, we parameterize and initialize the representation and membership in an efficient and effective way, as we will discuss in $\S$3.

Doubly Stochastic Subspace Clustering. To address the combinatorial nature of estimating the 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 ($\S$1.1). 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 $\Gamma \in \mathbb{R}^{n \times n}$ signaling the similarities between pairs of points, and then applies spectral clustering on the learned $\Gamma$ to obtain a final clustering [20, 22, 44, 41, 28, 75]. In particular, requiring doubly-stochastic constraints on the affinity $\Gamma$ is shown theoretically to suppress false inter-cluster connections for clustering [18] along with state-of-the-art empirical performance for subspace clustering [39].

Inspired by the above, we propose a constraint set $\Omega$ for matrix $\Gamma$ to be the set of $n \times n$ doubly stochastic matrices,

$$\Omega = \{ \Gamma \in \mathbb{R}^{n \times n} : \Gamma_{i\cdot} \geq 0, \Gamma_{\cdot j} = \Gamma_{\cdot j}^T \Gamma_{i\cdot} = 1 \} \quad (3)$$

However, this constraint alone is insufficient for strong clustering performance: Consider optimizing $\text{MCR}^2$ with respect to $\Gamma \in \Omega$ only, and note that the objective is convex with respect to $\Gamma$. Since we maximize a convex function with respect to convex constraints $\Omega$, an optimal $\Gamma$ would lie at an extreme point of $\Omega$, 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 use a negative entropy regularization $\sum_{ij} \Gamma_{ij} \log(\Gamma_{ij})$ to $\Gamma$ which biases $\Gamma$ toward the uniform matrix $\frac{1}{n} \mathbf{1} \mathbf{1}^T$. By tuning the coefficient of such regularization, we can also tune the sparsity level of $\Gamma$. This regularization can be conveniently integrated into the network architecture, as we will see in $\S$3. Now we are ready to state our proposed formulation Manifold Linearizing and Clustering ($\text{MLC}$):

$$\max_{\theta} R(Z; \epsilon) - R_c(Z, \Gamma; \epsilon) \quad \text{s.t.} \quad Z = f_\theta(X), \Gamma = h_\theta(X) \in \Omega, \quad (4)$$

where

$$R_c(Z, \Gamma; \epsilon) = \frac{1}{n} \sum_{j=1}^n \log \det \left( I + \frac{d}{\epsilon^2} Z \text{Diag}(\Gamma_{ij}) Z^T \right).$$

Note that both $Z$ and $\Gamma$ are parameterized by neural networks. While a doubly stochastic membership $\Gamma$ may seem large in size and hard to constrain, we explain in $\S$3 how we parameterize $\Gamma$ so that the constraints are satisfied by construction and efficient mini-batching is allowed.

Comparison with $\text{NMCE}$. A recent paper ($\text{NMCE}$) [38] studies the same Problem 1 as in this paper, and also proposes to optimize $\text{MCR}^2$ over both the representation and membership. Their method adopts an $n \times k$ matrix $\Pi$ to model the point-cluster membership; in contrast $\text{MLC}$ uses a doubly stochastic point-point membership $\Gamma$ inspired from the state-of-the-art subspace clustering (as stated above). Although seemingly not particularly significant, we note that in practice this allows for significantly simpler initialization strategies since we can initialize with an $n \times n$ estimate of an affinity matrix rather than a $n \times k$ estimate of cluster membership. We further elaborate on algorithmic differences at
the end of §3.2, and give empirical evidence that MLC is more accurate (Table 3) and stable against randomness (§F).

3. Algorithm

In this section, we describe how to parameterize the representation $Z$ and doubly stochastic membership $\Gamma$ (§3.1), as well as how to initialize them (§3.2) – in an efficient and effective manner. We summarize the meta-algorithm in Algorithm 1, and the overall architecture in Figure 2.

3.1. Efficient Parameterization

**Parameterizing $Z$.** As is common practice, we take an existing network architecture such as ResNet-18 as the backbone. We append a few affine layers with non-linearities as the *feature head* to further transform the output of the backbone to $\mathbb{R}^d$, followed by a projection layer to respect the unit sphere $S^{d-1}$ constraint.

**Parameterizing $\Gamma$.** Different from parameterizing $Z$, this is much less trivial: If one were to directly take $\Gamma$ as decision variables in $\Omega$, it would lead to maintaining $O(n^2)$ variables, which is prohibitive for large datasets (e.g., $n = 10^6$ for ImageNet). To allow efficient computation, we again draw inspiration from *subspace clustering*: There, the membership $\Gamma$ given data $X$ often takes the form of $g(X)^T g(X)$ for some transformation $g$, such as in the inner product kernel [28, 18] where $g(X) = X$ or the least square regression [44] where $g(X) = (I + \lambda X^T X)^{-1/2} X$. It is then tempting to take a neural network $g_\theta$ and use $C^\top C$ as the membership where $C = g_\theta(X)$. Nevertheless, such a matrix is in general not doubly stochastic, i.e., $C^\top C \notin \Omega$. To obtain a doubly stochastic membership, we further apply a Sinkhorn projection layer $P_{\Omega, \eta} (\cdot)$ [56, 19], which gives $\Gamma = P_{\Omega, \eta}(C^\top C) \in \Omega$, where $\eta$ is the coefficient of entropy regularization.\(^8\) As in parameterizing $Z$, we implement $g_\theta$ by taking the same backbone and appending layers of the same type to be the cluster head. As we shall see soon in §3.2, such a parameterization further allows us to initialize both $Z$ and $\Gamma$ in one shot using self-supervised learning.

**Complexity.** Thanks to the above parameterization, we can do forward and backward passes efficiently via mini-batches. For a mini-batch of $n_b$ samples ($n_b \ll n$ typically), the mini-batched versions of $Z$, $C$ and $\Gamma$ have sizes $d \times n_b$, $d \times n_b$ and $n_b \times n_b$, respectively (Figure 2).\(^9\)

3.2. Efficient Initialization

Since the proposed MLC objective (4) is non-convex, it is important to properly initialize both $Z$ and $\Gamma$ to converge to good (local) minimum.

**Initializing $Z$: Self-supervised Representation Learning.** Randomly initialized features could be far from being ideal (in the sense defined in §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 initialize the features using a self-supervised learning called *total coding rate* (TCR) [38]

$$\max_{\theta} R\left(\frac{Z + Z'}{2} ; \epsilon\right) + \lambda \sum_{i=1}^{n} |z_i^T z_i'|,$$

s.t. $z_i', z_i \in \mathbb{S}^{d-1}$, $\forall i \in [n]$, \(^5\)

where for every $i$, $z_i$ and $z_i'$ 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.\(^10\)

---

\(^8\)So in (4), $\Gamma = h_\theta(X)$ is simply $\Gamma = P_{\Omega, \eta}(g_\theta(X)^T g_\theta(X))$.  
\(^9\)Interested readers are referred to [3] for efficient computation of log det via variational forms, and [19] for efficient Sinkhorn iterations via implicit differentiation; these are beyond the scope of this paper.  
\(^10\)TCR is introduced here since it also uses coding rate measures. In
Initializing $\Gamma$. An ideal initialization of $\Gamma$ would be such that if $\Gamma_{ij}$ has a high value then points $i, j$ are likely to be from the same true cluster and vice versa. Luckily, after the self-supervised feature initialization mentioned above, $Z$ already have some structures which we can utilize. Thus, we propose to initialize $\Gamma$ with $P_{1:n}(Z^TZ)$; this is easily implemented by copying the parameters from the feature head to the cluster head once after the self-supervised initialization of the features, thanks to the parameterization of $\Gamma$ discussed in §3.1.

With the parameterization and initialization of our doubly stochastic membership $\Gamma$ set up, we are ready to contrast it with a popular alternative in the sequel.

Doubly Stochastic Membership vs. Point-Cluster Membership. Different from the doubly stochastic point-point membership $\Gamma$ proposed in this paper, prior deep representation learning and clustering works [64, 38, 48] often model a point-cluster membership. That is, an $n \times k$ matrix $\Pi$ where each row represents the probability of a point belonging to $k$ clusters. $\Pi$ is parameterized by a neural network (or a cluster head if one wishes), initialized randomly or otherwise via an extra training stage after the representation $Z$ is initialized. We highlight a few advantages of using a doubly stochastic point-point membership over a point-cluster one:

- **Stable**: As $\Gamma$ is initialized deterministically, the performance of $MLC$ is more stable compared to incorporating randomness in initializing the cluster head. We further justify this point empirically in §F.

- **Structured**: Initialization of $\Gamma$ takes advantage of structures from self-supervised initialized $Z$. As a side benefit, $MLC$ automatically gains from developments of self-supervised representation learning.

- **Efficient**: Once $Z$ is initialized, $\Gamma$ can be initialized with no additional cost whatsoever, compared to using any extra training stage to initialize the cluster head. In contrast, e.g., SCAN [64] trains the cluster head with 10 different random initializations, which is time-consuming.

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

$$Z = P_{d-1} \left( \frac{1}{A} \sum_{a=1}^A Z^{(a)} \right), \quad \Gamma = \frac{1}{A} \sum_{a=1}^A \Gamma^{(a)} \in \Omega. \quad (6)$$

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

4. Experiments on Real-World Datasets

We empirically verify that $MLC$ learns a union-of-orthogonal-subspaces representation, and yields more accurate clustering than state-of-the-art subspace clustering methods (§4.1). Further, we show that $MLC$ outperforms state-of-the-art deep clustering methods, even when there are many or imbalanced clusters (§4.2).

Metrics. To evaluate the clustering quality, we run spectral clustering on learned membership matrix $\Gamma$, and report the normalized mutual information (NMI, [60]) and clustering accuracy (ACC, [35]), as are commonly used in clustering tasks. To evaluate the learned representation, we define the following metric: for a collection of points $W = [w_1, \ldots, w_l] \in \mathbb{R}^{d \times l}$ (l > d) with associated singular values $\{\sigma_i\}_{i=1}^d$, define the numerical rank of $W$ 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 representation $Z$, as well as that of each ground-truth cluster of $Z$. A low numerical rank of $W$ implies that points in $W$ lie close to a low-dimensional subspace. We further report the cosine similarity of learned representation, which is simply $|z_i^T z_j|$ for points $i$ and $j$, since $||z_i|| = 1$ by construction in (4).

Finally, to compare the efficiency of methods we report the training time in §4.2, where the experiments are run on 2 Nvidia RTX3090 GPUs.

4.1. Comparison with Subspace Clustering

To demonstrate the ability of $MLC$ to cluster the samples and linearize the manifolds, we conduct experiments on

---

They are defined by the true membership, so that the numerical rank metric is decoupled from the quality of learned membership $\Gamma$. 

---

5455
CIFAR-10, which consists of RGB images from 10 classes such as planes, birds, and deer. 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 CIFAR-10 will yield low clustering accuracy. In contrast, MLC is able to achieve high clustering accuracy, and moreover, produce a union-of-orthogonal-subspaces representation on which subspace clustering methods can also achieve high accuracy.

**Data.** We use the training split of CIFAR-10 containing 50000 RGB images, each of size $3 \times 32 \times 32$. We use the augmentation specified in the Appendix to perform self-supervised representation learning via TCR (5) and get $Z_{TCR}$. For a fair comparison, the so-learned $Z_{TCR}$ are used both as initialization for MLC (line 1 of Algorithm 1), and as the input for subspace clustering methods\(^{12}\). In MLC, for each image in each batch we randomly sample $A = 2$ augmentations to apply to the image. As an additional comparison, we also run subspace clustering methods on the features $Z_{MLC}$ learned by MLC.

**Methods.** We compare with the elastic-net subspace clustering with active-set solver (EnSC, [75]) and sparse subspace clustering with orthogonal matching pursuit solver (SSC-OMP, [76]), using off-the-shelf implementation provided by the authors\(^{13}\). 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 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 [27]. 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.

\(^{12}\)Self-supervised features learned via coding rate measures empirically exhibit some union-of-subspace structure (one subspace per cluster). They have been used for subspace clustering as in, e.g., [78, §3.2] and [81, §4.2].

\(^{13}\)https://github.com/ChongYou/subspace-clustering.
Last but not 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.

Now we compare MLC with subspace clustering methods. Table 2 reports clustering accuracy and normalized mutual information for applying EnSC, SSC-OMP on self-supervised features $Z_{\text{TCR}}$, features $Z_{\text{MLC}}$ learned by MLC, and applying MLC on $X$, where $X$ is $6 \cdot 10^3$ images from 10 classes of CIFAR-10. 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 $X$, which surpasses EnSC (72.2%) and SSC-OMP (67.8%) on $Z_{\text{TCR}}$ by a large margin, even though $Z_{\text{TCR}}$ is used both as initialization for MLC and input for EnSC and SSC-OMP. Interestingly, using instead the features $Z_{\text{MLC}}$ 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 $Z_{\text{MLC}}$ has a union-of-subspace structure that can be utilized by subspace clustering. 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.

### 4.2. Comparison with Deep Clustering

We further compare the proposed MLC with state-of-the-art deep clustering methods on large-scale datasets (CIFAR-10, -20, -100, and TinyImageNet-200). Different than MLC, most methods reported (all except NMCE as discussed in §2.2) do not aim to learn a union-of-orthogonal-subspaces representation. Be that as it may, MLC achieves comparable or better clustering accuracy than state-of-the-art methods using less running time, even when the dataset presents many or imbalanced clusters.

**Datasets.** Beyond CIFAR-10 (§4.1), we further use CIFAR-20, CIFAR-100 and TinyImageNet-200 to evaluate the performance of our method. Both CIFAR-100 and CIFAR-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, TinyImageNet contains 100000 train images and 10000 test images with size $64 \times 64 \times 3$ split into 200 clusters.

**Baseline Methods.** We include clustering accuracy and normalized mutual information reported by SCAN [64], GCC [82], NNM [17], IMC [49], NMCE [38], SPICE [48] on aforementioned datasets whenever applicable. In addition, to compare the running time as well as to have more baseline methods when there are many clusters, we conduct experiments with MLC, SCAN [64], and IMC [49] on CIFAR-100. Training details are left to §B.2. For a fair comparison, all methods reported use ResNet-18 as backbone, commonly adopted by the literature. Note that each method chooses one or more pre-training that best fits its objective, such as SimCLR [12], SwAV [11], TCR [38], MoCoV2 [14]. Hence, for clarity, we indicate the pre-training used after the method, e.g., SCAN-SimCLR means SCAN method initialized with SimCLR.

**Results on CIFAR-10, -20.** Table 3 presents clustering accuracy and normalized mutual information of various methods. Overall, MLC is the most accurate, among

<table>
<thead>
<tr>
<th>Method vs. Dataset &amp; Metric</th>
<th>CIFAR-10 ACC</th>
<th>CIFAR-10 NMI</th>
<th>CIFAR-20 ACC</th>
<th>CIFAR-20 NMI</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCAN-SimCLR (ECCV '20)</td>
<td>0.876</td>
<td>0.787</td>
<td>0.468</td>
<td>0.459</td>
</tr>
<tr>
<td>GCC-SimCLR (ECCV '21)</td>
<td>0.856</td>
<td>0.764</td>
<td>0.472</td>
<td>0.472</td>
</tr>
<tr>
<td>NNM-SimCLR (CVRP '21)</td>
<td>0.843</td>
<td>0.748</td>
<td>0.477</td>
<td>0.484</td>
</tr>
<tr>
<td>IMC-SwAV (KBS '22)</td>
<td>0.891</td>
<td>0.811</td>
<td>0.490</td>
<td>0.503</td>
</tr>
<tr>
<td>NMCE-TCR (Arxiv '22)</td>
<td>0.830</td>
<td>0.761</td>
<td>0.437</td>
<td>0.488</td>
</tr>
<tr>
<td>MLC-TCR (Ours)</td>
<td>0.863</td>
<td>0.783</td>
<td>0.522</td>
<td>0.546</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method vs. Dataset &amp; Metric</th>
<th>CIFAR-10 ACC</th>
<th>CIFAR-10 NMI</th>
<th>TinyImageNet-200 ACC</th>
<th>TinyImageNet-200 NMI</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCAN-SimCLR (ECCV '20)</td>
<td>0.874</td>
<td>0.786</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GCC-SimCLR (ECCV '21)</td>
<td>0.856</td>
<td>0.764</td>
<td>0.535</td>
<td>0.565</td>
</tr>
<tr>
<td>IMC-SwAV (KBS '22)</td>
<td>0.918</td>
<td>0.850</td>
<td>0.305</td>
<td>0.449</td>
</tr>
<tr>
<td>MLC-MoCoV2 (Ours)</td>
<td>0.922</td>
<td>0.855</td>
<td>0.583</td>
<td>0.596</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method vs. Metric &amp; Stage</th>
<th>Running Time</th>
<th>Total</th>
<th>ACC</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCAN-SimCLR (ECCV '20)</td>
<td>308.4</td>
<td>54.7</td>
<td>396.3</td>
</tr>
<tr>
<td>IMC-SwAV (KBS '22)</td>
<td>529.4</td>
<td>-</td>
<td>529.4</td>
</tr>
<tr>
<td>MLC-TCR (Ours)</td>
<td>266.7</td>
<td>17.7</td>
<td>284.4</td>
</tr>
</tbody>
</table>

Table 3: Clustering accuracy and normalized mutual information of different methods on CIFAR-10 and CIFAR-20. For a fair comparison, all methods use ResNet-18 as backbone.

Table 4: Clustering accuracy and normalized mutual information of different methods on CIFAR-100 and TinyImageNet-200. For a fair comparison, all methods use ResNet-18 as backbone.
methods using either the same pre-training (middle and bottom rows) or any pre-training. To begin with, using TCR as pre-training, MLC achieves 86.3% and 52.2% clustering accuracies on CIFAR-10 and -20, which are 3.3 and 8.5 percentage points higher respectively than those achieved by NMCE\textsuperscript{14}. Remarkably, when MLC is initialized with MoCoV2 pre-training, it yields even higher clustering accuracies of 92.2% on CIFAR-10 and 58.3% on CIFAR-20, surpassing previous state-of-the-art methods SPICE-MoCoV2 (91.8%, 53.5%) and IMC-SwAV (89.1%, 49.0%). Interestingly, while the clustering performance of MLC-TCR is competitive on CIFAR-20, it is less so on CIFAR-10. After investigation, we find that the learned clusters appear semantically meaningful, even though they do not agree with the ground-truth labels of CIFAR-10 used for evaluation; we leave the details to §A.

Results on CIFAR-100, TinyImageNet-200. We report clustering accuracy and normalized mutual information on both datasets in Table 4, and further show running time on CIFAR-100 in Table 5. Notably, MLC outperforms SCAN and IMC-SwAV on CIFAR-100 and TinyImageNet-200 by a large margin, while using lower running time. E.g., on CIFAR-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.

Results on Imbalanced Clusters. Note that for CIFAR-10 or CIFAR-100 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 CIFAR-10, we remove half of the samples from odd-numbered clusters (i.e., clusters 1, 3, . . . , 9) from both the training and test split. We refer to the reduced dataset Imb-CIFAR-10. Likewise we construct Imb-CIFAR-100. We run two state-of-the-art methods IMC-SwAV and SCAN as well as the proposed MLC on Imb-CIFAR-10 and Imb-CIFAR-100.

Table 6 shows clustering accuracy on the imbalanced datasets Imb-CIFAR-10 and Imb-CIFAR-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 CIFAR-10 to Imb-CIFAR-10 the accuracy of MLC drops from 86% to 80%, whereas that of SCAN and IMC-SwAV decreases from above 87% to below 60%.

5. Conclusion

This paper studies the problem of simultaneously clustering and learning an union-of-orthogonal-subspaces 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.

Acknowledgements This work was partially supported by ONR grant N00014-22-1-2102, the joint Simons Foundation-NSF DMS grant 2031899, a research grant from TBSI, NSF grant 1704458, the Northrop Grumman Mission Systems Research in Applications for Learning Machines (REALM) initiative, and NSF graduate fellowship DGE2139757.

References


[33] Connor Lane, Ron Boger, Chong You, Manolis Tsakiris, Benjamin Haeffele, and Rene Vidal. Classifying and comparing approaches to subspace clustering with missing data. In Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, pages 0–0, 2019. 1


