entropy

MDPI

Article

CTRL: Closed-Loop Transcription to an LDR via Minimaxing Rate Reduction

Xili Dai [1,2,†], Shengbang Tong [1,†], Mingyang Li [3,†], Ziyang Wu [4,†], Michael Psenka [1], Kwan Ho Ryan Chan [5], Pengyuan Zhai [6], Yaodong Yu [1], Xiaojun Yuan [2], Heung-Yeung Shum [4] and Yi Ma [1,3,*]

1 Department of EECS, University of California Berkeley, Berkeley, CA 94720, USA; daixili_cs@163.com (X.D.); tsb@berkeley.edu (S.T.); psenka@berkeley.edu (M.P.); yaodong_yu@berkeley.edu (Y.Y.)
2 School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 610056, China; xjyuan@uestc.edu.cn
3 Tsinghua-Berkeley Shenzhen Institute, Shenzhen 518055, China; lmy17@mails.tsinghua.edu.cn
4 International Digital Economy Academy, Shenzhen 518048, China; robinwuzy@gmail.com (Z.W.); msraharry@hotmail.com (H.-Y.S.)
5 Mathematical Institute for Data Science, Johns Hopkins University, Baltimore, MD 21218, USA; kchan49@jhu.edu
6 Institute for Applied Computational Science, Harvard University, Cambridge, MA 02138, USA; pzhai@g.harvard.edu
* Correspondence: yima@eecs.berkeley.edu
† These authors contributed equally to this work.

Abstract: This work proposes a new computational framework for learning a structured generative model for real-world datasets. In particular, we propose to learn *a **C**losed-loop **Tr**anscription*between a multi-class, multi-dimensional data distribution and a ***L**inear discriminative representation* (*CTRL*) in the feature space that consists of multiple independent multi-dimensional linear subspaces. In particular, we argue that the optimal encoding and decoding mappings sought can be formulated as a *two-player minimax game between the encoder and decoder*for the learned representation. A natural utility function for this game is the so-called *rate reduction*, a simple information-theoretic measure for distances between mixtures of subspace-like Gaussians in the feature space. Our formulation draws inspiration from closed-loop error feedback from control systems and avoids expensive evaluating and minimizing of approximated distances between arbitrary distributions in either the data space or the feature space. To a large extent, this new formulation unifies the concepts and benefits of Auto-Encoding and GAN and naturally extends them to the settings of learning a *both discriminative and generative* representation for multi-class and multi-dimensional real-world data. Our extensive experiments on many benchmark imagery datasets demonstrate tremendous potential of this new closed-loop formulation: under fair comparison, visual quality of the learned decoder and classification performance of the encoder is competitive and arguably better than existing methods based on GAN, VAE, or a combination of both. Unlike existing generative models, the so-learned features of the multiple classes are structured instead of hidden: different classes are explicitly mapped onto corresponding *independent principal subspaces* in the feature space, and diverse visual attributes within each class are modeled by the *independent principal components* within each subspace.

Keywords: closed-loop transcription; linear discriminative representation; rate reduction; minimax game

Citation: Dai, X.; Tong, S.; Li, M.; Wu, Z.; Psenka, M.; Chan, K.H.R.; Zhai, P.; Yu, Y.; Yuan, X.; Shum, H.-Y.; et al. CTRL: Closed-Loop Transcription to an LDR via Minimaxing Rate Reduction. *Entropy* **2022**, *24*, 456. https://doi.org/10.3390/e24040456

Academic Editors: Lizhong Zheng and Chao Tian

Received: 10 February 2022
Accepted: 17 March 2022
Published: 25 March 2022

1. Introduction

One of the most fundamental tasks in modern data science and machine learning is to learn and model complex distributions (or structures) of real-world data, such as images or texts, from a set of observed samples. By "to learn and model", one typically means that

we want to establish a (parametric) mapping between the distribution of the real data, say $\boldsymbol{x} \in \mathbb{R}^D$, and a more compact random variable, say $\boldsymbol{z} \in \mathbb{R}^d$:

$$f(\cdot, \theta) : \boldsymbol{x} \in \mathbb{R}^D \mapsto \boldsymbol{z} \in \mathbb{R}^d \quad \text{or the inverse} \quad g(\cdot, \eta) : \boldsymbol{z} \in \mathbb{R}^d \mapsto \boldsymbol{x} \in \mathbb{R}^D, \tag{1}$$

where $\boldsymbol{z}$ has a certain standard structure or distribution (e.g., normal distributions). The so-learned representation or feature $\boldsymbol{z}$ would be much easier to use for either generative (e.g., decoding or replaying) or discriminative (e.g., classification) purposes, or both.

Data embedding versus data transcription. *Be aware* that the support of the distribution of $\boldsymbol{x}$ (and that of $\boldsymbol{z}$) is typically *extremely low-dimensional* compared to that of the ambient space (for instance, the well-known CIFAR-10 datasets consist of RGB images with a resolution of 32×32. Despite the images being in a space of $\mathbb{R}^{3072}$, our experiments will show that the intrinsic dimension of each class is less than a dozen, even after they are mapped into a feature space of $\mathbb{R}^{128}$) hence the above mapping(s) may not be uniquely defined based on the support in the space $\mathbb{R}^D$ (or $\mathbb{R}^d$). In addition, the data $\boldsymbol{x}$ may contain multiple components (e.g., modes, classes), and the intrinsic dimensions of these components are not necessarily the same. Hence, without loss of generality, we may assume the data $\boldsymbol{x}$ to be distributed over a union of low-dimensional nonlinear submanifolds $\cup_{j=1}^{k} \mathcal{M}_j \subset \mathbb{R}^D$, where each submanifold $\mathcal{M}_j$ is of dimension $d_j \ll D$. Regardless, we hope the learned mappings f and g are (locally dimension-preserving) *embedding* maps [1], when restricted to each of the components $\mathcal{M}_j$. In general, the dimension of the feature space d needs to be significantly higher than all of these intrinsic dimensions of the data: $d > d_j$. In fact, it should preferably be higher than the sum of all the intrinsic dimensions: $d \geq d_1 + \cdots + d_k$, since we normally expect that the features of different components/classes can be made fully independent or orthogonal in $\mathbb{R}^d$. Hence, without any explicit control of the mapping process, the actual features associated with images of the data under the embedding could still lie on some arbitrary nonlinear low-dimensional submanifolds inside the feature space $\mathbb{R}^d$. The distribution of the learned features remains "latent" or "hidden" in the feature space.

So, for features of the learned mappings (1) to be truly convenient to use for purposes such as data classification and generation, the goals of learning such mappings should not only simply reduce the dimension of the data $\boldsymbol{x}$ from D to d but also determine explicitly and precisely how the mapped feature $\boldsymbol{z} = f(\boldsymbol{x})$ is distributed within the feature space $\mathbb{R}^d$, in terms of both its support and density. Moreover, we want to establish an explicit map $g(\cdot)$ from this distribution of feature $\boldsymbol{z}$ back to the data space such that the distribution of its image $\hat{\boldsymbol{x}} = g(\boldsymbol{z})$ (closely) matches that of $\boldsymbol{x}$. To differentiate from finding arbitrary feature embeddings (as most existing methods do), we call embeddings of data onto an explicit family of models (structures or distributions) in the feature space as *data transcription*.

Paper Outline. This work is to show how such transcription can be achieved for real-world visual data with one important family of models: the linear discriminative representation (LDR) introduced by [2]. Before we formally introduce our approach in Section 2, for the remainder of this section, we first discuss two existing approaches, namely autoencoding and GAN, that are closely related to ours. As these approaches are rather popular and known to the readers, we will mainly point out some of their main conceptual and practical limitations that have motivated this work. Although our objective and framework will be mathematically formulated, the main purpose of this work is to verify the effectiveness of this new approach empirically through extensive experimentation, organized and presented in Section 3 and Appendix A. Our work presents compelling evidence that the closed-loop data transcription problem and our rate-reduction-based formulation deserve serious attention from the information-theoretical and mathematical communities. This has raised many exciting and open theoretical problems or hypotheses about learning, representing, and generating distributions or manifolds of high-dimensional real-world data. We discuss some open problems in Section 4 and new directions in Section 5. Source code can be found at https://github.com/Delay-Xili/LDR (accessed on 9 February 2022).

1.1. Learning Generative Models via Auto-Encoding or GAN

Auto-Encoding and its variants. In the machine-learning literature, roughly speaking,there have been two representative approaches to such a distribution-learning task. One is the classic "Auto Encoding" (AE) approach [3,4] that aims to simultaneously learn an encoding mapping f from $\boldsymbol{x}$ to $\boldsymbol{z}$ and an (inverse) decoding mapping g from $\boldsymbol{z}$ back to $\boldsymbol{x}$:

$$\boldsymbol{X} \xrightarrow{f(\boldsymbol{x},\theta)} \boldsymbol{Z} \xrightarrow{g(\boldsymbol{z},\eta)} \hat{\boldsymbol{X}}. \tag{2}$$

Here, we use bold capital letters to indicate a matrix of finite samples $\boldsymbol{X} = [\boldsymbol{x}^1, \ldots, \boldsymbol{x}^n] \in \mathbb{R}^{D\times n}$ of $\boldsymbol{x}$ and their mapped features $\boldsymbol{Z} = [\boldsymbol{z}^1, \ldots, \boldsymbol{z}^n] \subset \mathbb{R}^{d\times n}$, respectively. Typically, one wishes for two properties: firstly, the decoded samples $\hat{\boldsymbol{X}}$ are "similar" or close to the original $\boldsymbol{X}$, say in terms of maximum likelihood $p(\boldsymbol{X})$; and secondly, the (empirical) distribution of the mapped samples $\boldsymbol{Z}$, denoted as $\hat{p}(\boldsymbol{z}|\boldsymbol{X})$, is close to certain desired prior distribution $p(\boldsymbol{z})$, say some much lower-dimensional multivariate Gaussian (The classical PCA can be viewed as a special case of this task. In fact, the original auto-encoding is precisely cast as *nonlinear* PCA [3], assuming the data lie on only one nonlinear submanifold $\mathcal{M}$).

However it is typically very difficult, often computationally intractable to maximize the likelihood function $p(\boldsymbol{X})$ or to minimize certain "distance", say the *KL-divergence* $\mathcal{D}_{KL}(\hat{p}, p)$, between $\hat{p}(\boldsymbol{z}|\boldsymbol{X})$ and $p(\boldsymbol{z})$. Except for simple distributions such as Gaussian, the KL divergence usually does not have a closed-form, even for a mixture of Gaussians. The likelihood and the KL-divergence become ill-conditioned when the supports of the distributions are low-dimensional (i.e., degenerate) and not overlapping (which is almost always the case in practice when dealing with distributions of high-dimensional data in high-dimensional spaces). So in practice, one typically chooses to minimize instead certain approximate bounds or surrogates derived with various simplifying assumptions on the distributions involved, as is the case in variational auto-encoding (VAE) [5,6]. As a result, even after learning, the precise posterior distribution of $\hat{p}(\boldsymbol{z}|\boldsymbol{X})$ remains unclear or hidden inside the feature space.

In this work, we will show that if we impose specific requirements on the (distribution of) learned feature $\boldsymbol{z}$ to be a mixture of subspace-like Gaussians, a natural closed-form distance can be introduced for such distributions based on rate distortion from the information theory. In addition, the optimal solution to the feature representation within this family can be learned directly from the data *without specifying any target* $p(\boldsymbol{z})$ *in advance*, which is particularly difficult in practice when the distribution of a mixed dataset is multi-modal and each component may have a different dimension.

GAN and its variants. Compared to measuring distribution distance in the (often controlled) feature space $\boldsymbol{z}$, a much more challenging issue with the above auto-encoding approach is how to effectively measure the distance between the decoded samples $\hat{\boldsymbol{X}}$ and the original $\boldsymbol{X}$ in the data space $\boldsymbol{x}$. For instance, for visual data such as images, their distributions $p(\boldsymbol{X})$ or generative models $p(\boldsymbol{X}|\boldsymbol{z})$ are often not known. Despite extensive studies in the computer vision and image processing literature [7], it remains elusive to find a good measure for similarity of real images that is both efficient to compute and effective in capturing visual quality and semantic information of the images equally well. Precisely due to such difficulties, it has been suggested early on by [8] that one may have to take a discriminative approach to learn the distribution or a generative model for visual data. More recently, *Generative Adversarial Nets (GAN)* [9] offers an ingenious idea to alleviate this difficulty by utilizing a powerful discriminator d, usually modeled and learned by a deep network, to discern differences between the generated samples $\hat{\boldsymbol{X}}$ and the real ones $\boldsymbol{X}$:

$$\boldsymbol{Z} \xrightarrow{g(\boldsymbol{z},\eta)} \hat{\boldsymbol{X}},\ \boldsymbol{X} \xrightarrow{d(\boldsymbol{x},\theta)} \mathbf{0}, \mathbf{1}. \tag{3}$$

To a large extent, such a discriminator plays the role of minimizing certain distributional distance, e.g., the *Jensen–Shannon divergence*, between the data $\boldsymbol{X}$ and $\hat{\boldsymbol{X}}$. Compared to the KL-divergence, the JS-divergence is well-defined even if the supports of the two

distributions are non-overlapping. (However, JS-divergence does not have a closed-form expression even between two Gaussians, whereas KL-divergence does). However, as shown in [10], since the data distributions are low-dimensional, the JS-divergence can be highly ill-conditioned to optimize. (This may explain why many additional heuristics are typically used in many subsequent variants of GAN). So, instead, one may choose to replace JS-divergence with the earth mover's distance or the Wasserstein distance. However both JS-divergence and W-distance can only be approximately computed between two general distributions. (For instance, the W-distance requires one to compute the maximal difference between expectations of the two distributions over all 1-Lipschitz functions). Furthermore, neither the JS-divergence nor the W-distance have closed-form formulae, even for the Gaussian distributions. (The (ℓ^1-norm) W-distance can be bounded by the (ℓ^2-norm) W2-distance which has a closed-form [11]. However, as is well-known in high-dimensional geometry, ℓ^1-norm and ℓ^2 norm deviate significantly in terms of their geometric and statistical properties as the dimension becomes high [12]. The bound can become very loose). However, from a data representation perspective, *subspace-like Gaussians (e.g., PCA) or a mixture of them are the most desirable family of distributions that we wish our features to become.* This would make all subsequent tasks (generative or discriminative) much easier. In this work, we will show how to achieve this with a different fundamental metric, known as the rate reduction, introduced by [13].

The original GAN aims to directly learn a mapping $g(\cdot)$, called a generator, from a standard distribution (say, a low-dimensional Gaussian random field) to the real (visual) data distribution in a high-dimensional space. However, distributions of real-world data can be rather sophisticated and often contain *multiple* classes and *multiple* factors in each class [14]. This makes learning the mapping g rather challenging in practice, suffering difficulties such as *mode-collapse* [15]. As a result, many variants of GAN have been subsequently developed in order to improve the stability and performance in learning multiple modes and disentangling different factors in the data distribution, such as *Conditional GAN* [16–20], *InfoGAN* [21,22], or *Implicit Maximum Likelihood Estimation (IMLE)* [23,24]. In particular, to learn a generator for multi-class data, prevalent conditional GAN literature requires label information as conditional inputs [16,25–27]. Recently, [28,29] has proposed training a k-class GAN by generalizing the two-class cross entropy to a $(k+1)$-class cross entropy. In this work, *we will introduce a more refined* 2k*-class measure* for the k real and k generated classes. In addition, to avoid features for each class collapsing to a singleton [30], instead of cross entropy, *we will use the so-called rate-reduction measure that promotes multi-mode and multi-dimension in the learned features* [13]. One may view the rate reduction as a metric distance that has closed-form formulae for a mixture of (subspace-like) Gaussians, whereas neither JS-divergence nor W-distance can be computed in closed form (even between two Gaussians).

Another line of research is about how to stabilize the training of GAN. SN-GAN [31] has shown that spectral normalization on the discriminator is rather effective, which we will adopt in our work, although our formulation is not so sensitive to such choice designed for GAN (see ablation study in Appendix A.9). PacGAN [32] shows that the training stability can be significantly improved by packing a pair of real and generated images together for the discriminator. Inspired by this work, *we show how to generalize such an idea to discriminating an arbitrary number of pairs of real and decoded samples without concatenating the samples.* Our results in this work will even suggest that the larger the batch size discriminated, the merrier (see ablation study in Appendix A.10). In addition, ref. [29] has shown that optimizing the latent features leads to state-of-the-art visual quality. Their method is based on the deep compressed sensing GAN [28]. Hence, there are strong reasons to believe that their method essentially utilizes the *compressed sensing* principle [12] to implicitly exploit the low-dimensionality of the feature distribution. Our framework *will explicitly expose and exploit such low-dimensional structures on the learned feature distribution.*

Combination of AE and GAN. Although AE (VAE) and GAN originated with somewhat different motivations, they have evolved into popular and effective frameworks for

learning and modeling complex distributions of many real-world data such as images. (In fact, in some idealistic settings, it can be shown that AE and GAN are actually equivalent: for instance, in the LOG settings, authors in [33] have shown that GAN coincides with the classic PCA, which is precisely the solution to auto-encoding in the linear case). Many recent efforts tend to combine both auto-encoding and GAN to generate more powerful generative frameworks for more diverse data sets, such as [15,34–42]. As we will see, in our framework, AE and GAN can be naturally interpreted as two different segments of a closed-loop data transcription process. However, unlike GAN or AE (VAE), the "origin" or "target" distribution of the feature $\boldsymbol{z}$ will no longer be specified *a priori*, and is instead learned from the data $\boldsymbol{x}$. In addition, *this intrinsically low-dimensional distribution of* $\boldsymbol{z}$ *(with all of its low-dimensional supports) is explicitly modeled as a mixture of orthogonal subspaces (or independent Gaussians) within the feature space* $\mathbb{R}^d$, sometimes known as the principal subspaces.

Universality of Representations. Note that GANs (and most VAEs) are typically designed without explicit modeling assumptions on the distribution of the data nor on the features. Many even believe that it is this "universal" distribution learning capability (assuming minimizing distances between arbitrary distributions in high-dimensional space can be solved efficiently, which unfortunately has many caveats and often is impractical) that is attributed to their empirical success in learning distributions of complicated data such as images. In this work, we will provide empirical evidence that such an "arbitrary distribution learning machine" might not be necessary. (In fact, it may be computationally intractable in general). A *controlled and deformed* family of low-dimensional linear subspaces (Gaussians) can be more than powerful, and expressive enough to model real-world visual data. (In fact, a Gaussian mixture model is already a universal approximator of almost arbitrary densities [43]. Hence, we do not loose any generality at all). As we will also see, once we can place a proper and precise metric on such models, the associated learning problems can become much better conditioned and more amenable to rigorous analysis and performance guarantees in the future.

1.2. Learning Linear Discriminative Representation via Rate Reduction

Recently, the authors in [2] proposed a new objective for deep learning that aims to learn a *linear discriminative representation* (LDR) for multi-class data. The basic idea is to map distributions of real data, potentially on *multiple* nonlinear submanifolds $\cup_{j=1}^k \mathcal{M}_j \subset \mathbb{R}^D$ (in classical statistical settings, such nonlinear structures of the data were also referred to as principal curves or surfaces [44,45]. There has been a long quest of trying to extend PCA to handle potential nonlinear low-dimensional structures in data distribution (see [46] for a thorough survey) to a family of canonical models consisting of multiple independent (or orthogonal) linear subspaces, denoted as $\cup_{j=1}^k \mathcal{S}_j \subset \mathbb{R}^d$. To some extent, this generalizes the classic nonlinear PCA [3] to more general/realistic settings where we simultaneously apply *multiple nonlinear PCAs* to data on multiple nonlinear submanifolds. Or equivalently, the problem can also be viewed as a nonlinear extension to the classic *Generalized PCA* (GPCA) [46]. (Conventionally, "generalized PCA" refers to generalizing the setting of PCA to multiple *linear* subspaces. Here, we need to further generalize multiple *nonlinear* submanifolds. Unlike conventional discriminative methods that only aim to predict class labels as one-hot vectors, the LDR aims to learn the likely multi-dimensional distribution of the data, hence it is suitable for both discriminative and generative purposes. It has been shown that this can be achieved via maximizing the so-called "rate reduction" objective based on the rate distortion of subspace-like Gaussians [47].

LDR via MCR^2. More precisely, consider a set of data samples $\boldsymbol{X} = [\boldsymbol{x}^1, \ldots, \boldsymbol{x}^n] \in \mathbb{R}^{D \times n}$ from k different classes. That is, we have $\boldsymbol{X} = \cup_{j=1}^k \boldsymbol{X}_j$ with each subset of samples $\boldsymbol{X}_j$ belonging to one of the low-dimensional submanifolds: $\boldsymbol{X}_j \subset \mathcal{M}_j, j = 1, \ldots, k$. Following the notation in [2], we use a matrix $\boldsymbol{\Pi}^j(i,i) = 1$ to denote the membership of sample i

belonging to class j (and $\mathbf{\Pi}^j = 0$ otherwise). One seeks a continuous mapping $f(\cdot, \theta) : \boldsymbol{x} \mapsto \boldsymbol{z}$ from $\boldsymbol{X}$ to an optimal representation $\boldsymbol{Z} = [\boldsymbol{z}^1, \ldots, \boldsymbol{z}^n] \subset \mathbb{R}^{d \times n}$:

$$\boldsymbol{X} \xrightarrow{f(\boldsymbol{x}, \theta)} \boldsymbol{Z}, \tag{4}$$

which maximizes the following coding rate-reduction objective, known as *the MCR*^{2} *principle* [13]:

$$\max_{\boldsymbol{Z}} \Delta R(\boldsymbol{Z} \,|\, \mathbf{\Pi}, \epsilon) \doteq \underbrace{\frac{1}{2} \log\det\left(\boldsymbol{I} + \alpha \boldsymbol{Z}\boldsymbol{Z}^*\right)}_{R(\boldsymbol{Z}\,|\,\epsilon)} - \underbrace{\sum_{j=1}^{k} \frac{\gamma_j}{2} \log\det\left(\boldsymbol{I} + \alpha_j \boldsymbol{Z}\mathbf{\Pi}^j\boldsymbol{Z}^*\right)}_{R_c(\boldsymbol{Z}\,|\,\mathbf{\Pi}, \epsilon)}, \tag{5}$$

where $\alpha = \frac{d}{n\epsilon^2}$, $\alpha_j = \frac{d}{\mathrm{tr}(\mathbf{\Pi}^j)\epsilon^2}$, $\gamma_j = \frac{\mathrm{tr}(\mathbf{\Pi}^j)}{n}$ for $j = 1, \ldots, k$. In this paper, for simplicity we denote $\Delta R(\boldsymbol{Z}\,|\,\mathbf{\Pi}, \epsilon)$ as $\Delta R(\boldsymbol{Z})$ assuming $\mathbf{\Pi}, \epsilon$ are known and fixed. The first term $R(\boldsymbol{Z}\,|\,\epsilon)$, or $R(\boldsymbol{Z})$ for short, is the coding rate of the whole feature set $\boldsymbol{Z}$ (coded as a Gaussian source) with a prescribed precision ϵ; the second term $R_c(\boldsymbol{Z}\,|\,\mathbf{\Pi}, \epsilon)$, or simply $R_c(\boldsymbol{Z})$, is the average coding rate of the k subsets of features $\boldsymbol{Z}_j = f(\boldsymbol{X}_j)$ (each coded as a Gaussian).

As has been shown by [13], maximizing the difference between the two terms will expand the whole feature set while compressing and linearizing features of each of the k classes. If the mapping f maximizes the rate reduction, it maps the features of different classes into independent (orthogonal) subspaces in $\mathbb{R}^d$. Figure 1 illustrates a simple example of data with $k = 2$ classes (on two submanifolds) mapped to two incoherent subspaces (solid black lines). Notice that, compared to AE (2) and GAN (3), the above mapping (4) is only one-sided: from the data $\boldsymbol{X}$ to the feature $\boldsymbol{Z}$. In this work, we will see how to use the rate-reduction metric to establish inverse mapping from the feature $\boldsymbol{Z}$ back to the data $\boldsymbol{X}$, while still preserving the subspace structures in the feature space.

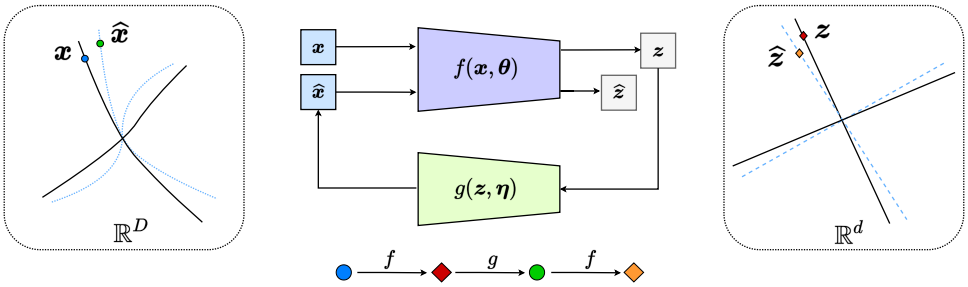


Figure 1. CTRL: A Closed-loop Transcription to an LDR. The encoder f has dual roles: it learns an LDR $\boldsymbol{z}$ for the data $\boldsymbol{x}$ via maximizing the rate reduction of $\boldsymbol{z}$ and it is also a "feedback sensor" for any discrepancy between the data $\boldsymbol{x}$ and the decoded $\hat{\boldsymbol{x}}$. The decoder g also has dual roles: it is a "controller" that corrects the discrepancy between $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$ and it also aims to minimize the overall coding rate for the learned LDR.

2. Data Transcription via Rate Reduction

2.1. Closed-Loop Transcription to an LDR (CTRL)

One issue with this one-sided LDR learning (4) is that maximizing the above objective (5) tends to expand the dimension of the learned subspace for features in each class (if the dimension of the feature space d is too high, maximizing the rate reduction may over-estimate the dimension of each class. Hence, to learn a good representation, one needs to pre-select a proper dimension for the feature space, as achieved in the experiments in [13]. In fact the same "model selection" problem persists even in the simplest single-subspace case, which is the classic PCA [48]. Selecting the correct number of principal components in a heterogeneous noisy situation remains an active research topic [49]). To verify whether the learned features are neither over-estimating nor under-estimating the data structure, we may consider learning a decoder $g(\cdot, \eta) : \boldsymbol{z} \mapsto \boldsymbol{x}$ from the representation $\boldsymbol{Z} = f(\boldsymbol{X}, \theta)$ back to the data space $\boldsymbol{x}$: $\hat{\boldsymbol{X}} = g(\boldsymbol{Z}, \eta)$, and check how close $\boldsymbol{X}$ and $\hat{\boldsymbol{X}}$ are or how close their features $\boldsymbol{Z}$

and $\hat{\boldsymbol{Z}} = f(\hat{\boldsymbol{X}}, \theta)$ are. In principle, the decoder g should examine if all the learned features by the encoder f are both necessary and sufficient for achieving this task. The overall pipeline can be illustrated by the following "closed-loop" diagram:

$$\boldsymbol{X} \xrightarrow{\quad f(\boldsymbol{x},\theta) \quad} \boldsymbol{Z} \xrightarrow{\quad g(\boldsymbol{z},\eta) \quad} \hat{\boldsymbol{X}} \xrightarrow{\quad f(\boldsymbol{x},\theta) \quad} \hat{\boldsymbol{Z}}, \tag{6}$$

where the overall model has parameters: $\Theta = \{\theta, \eta\}$.

Notice that in the above process, the segment from $\boldsymbol{X}$ to $\hat{\boldsymbol{X}}$ resembles a typical *Auto-Encoding* process; although, as we will soon see, our MCR^2-based encoder f plays an additional role as a discriminator. The segment from $\boldsymbol{Z}$ to $\hat{\boldsymbol{Z}}$ draws resemblance to the typical GAN process; although, in our context, the distribution of the latent variable $\boldsymbol{z}$ will be learned from the data $\boldsymbol{x}$. Despite these connections, as we will soon see, this new closed-loop formulation will allow us to utilize the *error feedback* mechanism (widely practiced in control systems) and directly enforce loop consistency between encoding and decoding (networks) *without* using any additional discriminator(s) that are typically needed in existing VAE/GAN architectures.

Here, in the specific context of rate reduction, we name this special auto-encoding process "*Transcription to an LDR*" since the maximal rate-reduction principle explicitly transcribes the data $\boldsymbol{X}$, via f, to features $\boldsymbol{Z}$ on a linear discriminative representation (LDR) (through our extensive experiments on diverse real-world visual datasets, one does not lose any generality or expressiveness by restricting to this special but rich class of models. On the contrary, the restriction significantly simplifies and improves the learning process), which can be subsequently decoded back to the data space $\hat{\boldsymbol{X}}$, via g. Hence, the encoding and decoding maps f and g together form a "closed-loop" process, as illustrated in Figure 1. We hope that this closed-loop transcription to an LDR (CTRL) has the following good properties:

- **Injectivity:** the generated $\hat{\boldsymbol{x}} = g(f(\boldsymbol{x}, \theta), \eta) \in \hat{\boldsymbol{X}}$ should be as close to (ideally the same as) the original data $\boldsymbol{x} \in \boldsymbol{X}$, in terms of certain measures of similarity or distance.
- **Surjectivity:** for all mapped images $\boldsymbol{z} = f(\boldsymbol{x}) \in \boldsymbol{Z}$ of the training data $\boldsymbol{x} \in \boldsymbol{X}$, there are decoded samples $\hat{\boldsymbol{z}} = f(g(\boldsymbol{z}, \eta), \theta) \in \hat{\boldsymbol{Z}}$ close to (ideally the same as) $\boldsymbol{z}$.

Mathematically, we seek an *embedding* of the data $\boldsymbol{x}$ supported on certain nonlinear submanifolds $\cup_{j=1}^{k} \mathcal{M}_j$ in the space $\mathbb{R}^D$ to feature $\boldsymbol{z}$ on a set of (discriminative) linear subspaces $\cup_{j=1}^{k} \mathcal{S}_j$ in the feature space $\mathbb{R}^d$. Ideally, both f and g should be embeddings [1], when restricted on the support of the data distribution or that of the features. (That is, we hope $f\,|_{\mathcal{M}_j}$ and $g\,|_{\mathcal{S}_j}$ are all embeddings for all $j = 1, \ldots, k$.) In addition, more ideally, we hope f and g are mutually inverse embeddings: $g \circ f = \mathrm{Id}$ (when restricted on the submanifolds). Nevertheless, if we are only interested in learning the distribution, embeddings of the support would often suffice the purposes (e.g., classification or generative purposes). Notice that the above goals are similar to many VAE+GAN-related methods in the machine-learning literature, such as BiGAN [38] and ALI [39]. We will discuss the differences of our approach from these existing methods in Section 2.3 (as well as providing some experimental comparisons in the Appendix A).

At first sight, this is a rather daunting task, since we are trying to learn over a (seemingly infinite-dimensional) functional space of all embeddings and distributions from finite samples. In this work, we will take a more pragmatic approach and show how one can learn a good encoding, decoding, and representation tuple: $(f, g, \boldsymbol{z})$ from $\boldsymbol{X}$ via tractable computational means. In particular, we will convert the above goals to certain feasible programs that optimize a sensible measure of goodness for the learned representations $\boldsymbol{Z}$.

2.2. Measuring Distances in the Feature Space and Data Space

Contractive measure for the decoder. For the *second* item in the above wishlist, as the representations in the feature space $\boldsymbol{z}$ are by design linear subspaces or (degenerate) Gaussians, we have geometrically or statistically meaningful metrics for both samples and

distributions in the feature space $\boldsymbol{z}$. For example, we care about the distance between distributions between the features of the original data $\boldsymbol{Z}$ and the transcribed $\hat{\boldsymbol{Z}}$. Since the features of each class, $\boldsymbol{Z}_j$ and $\hat{\boldsymbol{Z}}_j$, are similar to subspaces/Gaussians, their "distance" can be measured by the rate reduction, with (5) restricted to two sets of equal size:

$$\Delta R(\boldsymbol{Z}_j, \hat{\boldsymbol{Z}}_j) \doteq R(\boldsymbol{Z}_j \cup \hat{\boldsymbol{Z}}_j) - \frac{1}{2}\big(R(\boldsymbol{Z}_j) + R(\hat{\boldsymbol{Z}}_j)\big). \tag{7}$$

According to the interpretation of the rate reduction given in [13], the above quantity precisely measures the volume of the space between $\boldsymbol{Z}_j$ and $\hat{\boldsymbol{Z}}_j$, illustrated as a pair of black and blue lines in Figure 1. Then, for the "distance" of all, say k, classes, we simply sum the rate reduction for all pairs:

$$d(\boldsymbol{Z}, \hat{\boldsymbol{Z}}) \doteq \min_{\eta} \sum_{j=1}^{k} \Delta R(\boldsymbol{Z}_j, \hat{\boldsymbol{Z}}_j) = \min_{\eta} \sum_{j=1}^{k} \Delta R\big(\boldsymbol{Z}_j, f(g(\boldsymbol{Z}_j, \eta), \theta)\big), \tag{8}$$

where $\boldsymbol{Z}_j = f(\boldsymbol{X}_j, \theta)$ and $\hat{\boldsymbol{Z}}_j = f(\hat{\boldsymbol{X}}_j, \theta)$. Obviously, a main goal of the learned decoder $g(\cdot, \eta)$ is to *minimize* the distance between these distributions. Notice that if the encoder f preserves (i.e., injective for) the intrinsic structures of the original data $\boldsymbol{X}$, (this is typically the case for MCR^2-based feature representation [13]) this criterion essentially aims to ensure there will be some decoded sample $\hat{\boldsymbol{x}}$ close to every data sample $\boldsymbol{x}$—hence the decoder g should be "surjective". According to the ideas of IMLE [23], such a requirement could effectively help to avoid mode-collapsing or mode-dropping.

Contrastive measure for the encoder. For the *first* item in our wishlist, however, we normally do not have a natural metric or "distance" for similarity of samples or distributions in the original data space $\boldsymbol{x}$ for data such as images. As mentioned before, finding proper metrics or distance functions on natural images has always been an elusive and challenging task [7]. To alleviate this difficulty, we can measure the similarity or difference between $\hat{\boldsymbol{X}}$ and $\boldsymbol{X}$ through their mapped features $\hat{\boldsymbol{Z}}$ and $\boldsymbol{Z}$ in the feature space (again assuming f is structure-preserving). If we are interested in discerning *any* differences in the distributions of the original and transcribed samples, we may view the MCR^2 feature encoder $f(\cdot, \theta)$ as a "discriminator" to *magnify* any difference between all pairs of $\boldsymbol{X}_j$ and $\hat{\boldsymbol{X}}_j$, by simply maximizing, instead of minimizing, the *same quantity* in (8):

$$d(\boldsymbol{X}, \hat{\boldsymbol{X}}) \doteq \max_{\theta} \sum_{j=1}^{k} \Delta R(\boldsymbol{Z}_j, \hat{\boldsymbol{Z}}_j) = \max_{\theta} \sum_{j=1}^{k} \Delta R\big(f(\boldsymbol{X}_j, \theta), f(\hat{\boldsymbol{X}}_j, \theta)\big). \tag{9}$$

That is, a "distance" between $\boldsymbol{X}$ and $\hat{\boldsymbol{X}}$ can be measured as the maximally achievable rate reduction between all pairs of classes in these two sets. In a way, this measures how well or badly the decoded $\hat{\boldsymbol{X}}$ aligns with the original data $\boldsymbol{X}$—hence measuring the goodness of "injectivity" of the encoder f. Notice that such a discriminative measure is consistent with the idea of GAN [9] that tries to separate $\boldsymbol{X}$ and $\hat{\boldsymbol{X}}$ into two classes, measured by the cross-entropy. Nevertheless, here the MCR^2-based discriminator f naturally generalizes to cases when the data distributions are multi-class and multi-modal, and the discriminativeness is measured with a more refined measure—the rate reduction—instead of the typical two-class loss (e.g., cross entropy) used in GANs. See Appendix A.8 for comparisons with some ablation studies.

One may wonder why we need the mapping $f(\cdot, \theta)$ to function as a discriminator between $\boldsymbol{X}$ and $\hat{\boldsymbol{X}}$ by maximizing $\max_\theta \Delta R\big(f(\boldsymbol{X}, \theta), f(\hat{\boldsymbol{X}}, \theta)\big)$. Figure 2 gives a simple illustration: there might be many decoders g such that $f \circ g$ is an identity (Id) mapping. Here, we use the notion of "identity mapping" in a loose sense: depending on the context, it could simply mean an embedding from $S_{\boldsymbol{z}}$ to $S_{\boldsymbol{z}}$. $f \circ g(\boldsymbol{z}) = \boldsymbol{z}$ for all $\boldsymbol{z}$ in the subspace $S_{\boldsymbol{z}}$ in the feature space. However, $g \circ f$ is not necessarily an auto-encoding map for $\boldsymbol{x}$ in the original distribution $S_{\boldsymbol{x}}$ (here for simplicity drawn as a subspace). That is, $g \circ f(S_{\boldsymbol{x}}) \not\subset S_{\boldsymbol{x}}$, let alone $g \circ f(S_{\boldsymbol{x}}) = S_{\boldsymbol{x}}$ or $g \circ f(\boldsymbol{x}) = \boldsymbol{x}$. One should expect, without careful control of the

image of g, with high probability, this would be the case, especially when the support of the distribution of $\boldsymbol{x}$ is extremely low-dimensional in the original high-dimensional data space. For example, as we will see in the experiments, the intrinsic dimension of the submanifold associated with each image category is about a dozen, whereas images are embedded in a (pixel) space of thousands or tens of thousands of dimensions.

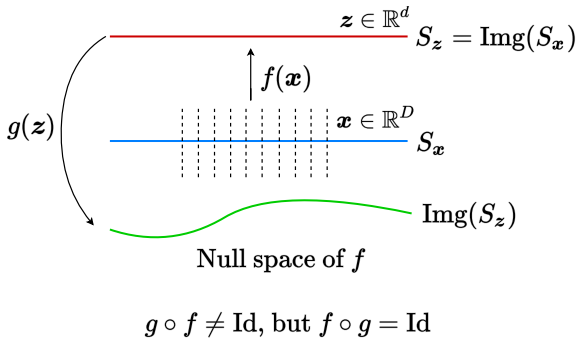


Figure 2. Embeddings of Low-Dimensional Submanifolds in High-Dimensional Spaces. $S_{\boldsymbol{x}}$ (blue) is the submanifold for the original data $\boldsymbol{x}$; $S_{\boldsymbol{z}}$ (red) is the image of $S_{\boldsymbol{x}}$ under the mapping f, representing the learned feature $\boldsymbol{z}$; and the green curve is the image of the feature $\boldsymbol{z}$ under the decoding mapping g.

Remark: representing the encoding and decoding mappings. Some practical questions arise immediately: how rich should the families of functions be that we should consider to use for the encoder f and decoder g that can optimize the above rate-reduction-type objectives? In fact, similar questions exist for the formulation of GAN, regarding the realizability of the data distribution by the generator, see [50]. Conceptually, here we know that the encoder f needs to be rich enough to discriminate (small) deviations from the true data support $\mathcal{M}_j$, while the decoder g needs to be expressive enough to generate the data distribution from the learned mixture of subspace-Gaussians. How should we represent or parameterize them, hence making our objectives computable and optimizable? For the most general cases, these remain widely open and challenging mathematical and computational problems. As we mentioned earlier, in this work, we will take a more pragmatic approach by simply representing these mappings with popular neural networks that have empirically proven to be good at approximating distributions of practical (visual) datasets or for achieving the maximum of the rate-reduction-type objectives [13]. Nevertheless, our experiments indicate that our formulation and objectives are *not so sensitive* to particular choices in network structures or many of the tricks used to train them. In addition, in the special cases when the real data distribution is benignly deformed from an LDR, the work of [2] has shown that one can explicitly construct these mappings from the rate-reduction objectives in the form of a deep network known as ReduNet. However, it remains unclear how such constructions could be generalized to closed-loop settings. Regardless, answers to these questions are beyond the scope of this work, as our purposes here are mainly to empirically verify the validity of the proposed closed-loop data transcription framework.

2.3. Encoding and Decoding as a Two-Player MiniMax Game

Comparing the contractive and contrastive nature of (8) and (9) on the same utility, we see the roles of the encoder $f(\cdot, \theta)$ and the decoder $g(\cdot, \eta)$ naturally as "**a two-player game**": *while the encoder f tries to magnify the difference between the original data and their transcribed data, the decoder g aims to minimize the difference.* Now for convenience, let us define the "closed-loop encoding" function:

$$h(\boldsymbol{x}, \theta, \eta) \doteq f\big(g\big(f(\boldsymbol{x}, \theta), \eta\big), \theta\big) : \ \boldsymbol{x} \mapsto \boldsymbol{z}. \tag{10}$$

Ideally, we want this function to be very close to $f(\boldsymbol{x}, \theta)$ or at least the distributions of their images should be close. With this notation, combining (8) and (9), a closed-loop notion of "distance" between $\boldsymbol{X}$ and $\hat{\boldsymbol{X}}$ can be computed as *an equilibrium point* to the following

Min-Max (or Max-Min) program for the same utility in terms of rate reduction (theoretically, there might be significant difference in formulating and seeking the desired solution as the equilibrium point to a min-max or max-min game. In practice, we do not see major differences as we optimize the program by simply alternating between minimization and maximization. We leave a more careful investigation to future work):

$$\mathcal{D}(\boldsymbol{X}, \hat{\boldsymbol{X}}) \doteq \min_{\eta} \max_{\theta} \sum_{j=1}^{k} \Delta R\big(f(\boldsymbol{X}_j, \theta), h(\boldsymbol{X}_j, \theta, \eta)\big). \tag{11}$$

Notice that this only measures the difference between (features of) the original data and its transcribed version. It does not measure how good the representation $\boldsymbol{Z}$ (or $\hat{\boldsymbol{Z}}$) is for the multiple classes within $\boldsymbol{X}$ (or $\hat{\boldsymbol{X}}$). To this end, we may combine the above distance with the original MCR^2-type objectives (5): namely, the rate reduction $\Delta R(\boldsymbol{Z})$ and $\Delta R(\hat{\boldsymbol{Z}})$ for the learned LDR $\boldsymbol{Z}$ for $\boldsymbol{X}$ and $\hat{\boldsymbol{Z}}$ for the decoded $\hat{\boldsymbol{X}}$. Notice that although the encoder f tries to *maximize* the multi-class rate reduction of the features $\boldsymbol{Z}$ of the data $\boldsymbol{X}$, the decoder g should *minimize* the rate reduction of the multi-class features $\hat{\boldsymbol{Z}}$ of the decoded $\hat{\boldsymbol{X}}$. That is, the decoder g tries to use a minimal coding rate needed to achieve a good decoding quality.

Hence, the overall "multi-class" Min-Max program for learning the Closed-loop Transcription to an LDR, named CTRL-Multi, is subject to certain constraints (upper or lower bounds) on the first term and the second term. In this work, we only consider the simple case by adding these rate-reduction quantities together. Of course, in the future, one may consider other more delicate formulations. For instance, we may consider a Min-Max game on the third term (11). Such constrained minimax games have also started to draw attention lately [51].

$$\begin{aligned}\min_{\eta} \max_{\theta} \mathcal{T}_{\boldsymbol{X}}(\theta, \eta) \;&\doteq\; \underbrace{\Delta R\big(f(\boldsymbol{X}, \theta)\big)}_{\text{Expansive encode}} + \underbrace{\Delta R\big(h(\boldsymbol{X}, \theta, \eta)\big)}_{\text{Compressive decode}} + \sum_{j=1}^{k} \underbrace{\Delta R\big(f(\boldsymbol{X}_j, \theta), h(\boldsymbol{X}_j, \theta, \eta)\big)}_{\text{Contrastive encode \& Contractive decode}} \\ &= \Delta R\big(\boldsymbol{Z}(\theta)\big) + \Delta R\big(\hat{\boldsymbol{Z}}(\theta, \eta)\big) + \sum_{j=1}^{k} \Delta R\big(\boldsymbol{Z}_j(\theta), \hat{\boldsymbol{Z}}_j(\theta, \eta)\big).\end{aligned} \tag{12}$$

Empirically, we have evaluated the necessity of these terms in an ablation study (see Appendix A.8.3). Notice that, without the terms associated with the generative part h or with all such terms fixed as constant, the above objective is precisely the original MCR^2 objective proposed by [13]. In an unsupervised setting, if we view each sample (and its augmentations) as its own class, the above formulation remains exactly the same. The number of classes k is simply the number of independent samples. In addition, notice that the minimax objective function depends only on (features of) the data $\boldsymbol{X}$, hence one can learn the encoder and decoder (parameters) without the need for sampling or matching any additional distribution (as typically needed in GANs or VAEs).

As a special case, if $\boldsymbol{X}$ only has one class, the above Min-Max program reduces (as the first two rate reduction terms automatically become zero) to a special "two-class" or "binary" form, named CTRL-Binary, between $\boldsymbol{X}$ and the decoded $\hat{\boldsymbol{X}}$ by viewing $\boldsymbol{X}$ and $\hat{\boldsymbol{X}}$ as two classes $\{\mathbf{0}, \mathbf{1}\}$. Notice that this binary case resembles formulation of the original GAN (3). Nevertheless, instead of using cross entropy, our formulation adopts a more refined rate-reduction measure, which has been shown to promote diversity in the learned representation [13]).

$$\text{CTRL-Binary:} \quad \min_{\eta} \max_{\theta} \mathcal{T}^{b}_{\boldsymbol{X}}(\theta, \eta) \doteq \Delta R\big(f(\boldsymbol{X}, \theta), h(\boldsymbol{X}, \theta, \eta)\big) = \Delta R\big(\boldsymbol{Z}(\theta), \hat{\boldsymbol{Z}}(\theta, \eta)\big). \tag{13}$$

Sometimes, even when $\boldsymbol{X}$ contains multiple classes/modes, one could still view all classes together as one class. Then, the above binary objective is to align the union distribution of all classes with their decoded $\hat{\boldsymbol{X}}$. This is typically a simpler task to achieve

than the multi-class one (12), since it does not require learning of a more refined multi-class CTRL for the data, as we will later see in experiments. Notice that one good characteristic of the above formulation is that *all quantities in the objectives are measured in terms of rate reduction for the learned features* (assuming features eventually become subspace Gaussians).

In all of our subsequent experiments, we solve the above minimax programs using the most basic gradient descent–ascent (GDA) algorithm [52] that alternates between the minimization and maximization, with the same learning rate and without any timescale separation (as typically needed for training GANs [53]). Although more refined optimization schemes can likely further improve the efficiency and performance, we leave these for future investigations.

Remark: closed-loop error correction. One may notice that our framework (see Figure 1) draws inspiration from closed-loop error correction widely practiced in feedback control systems. In the machine-learning and deep-learning literature, the idea of closed-loop error correction and closed-loop fixed point has been explored before to interpret the recursive error-correcting mechanism and explain stability in a forward (predictive) deep neural network, for example the *deep equilibrium networks* [54] and the *deep implicit networks* [55], again drawing inspiration from feedback control. Here, in our framework, the closed-loop mechanism is not used to interpret the encoding or decoding (forward) networks f and g. Instead, it is used to form an overall feedback system between the two encoding and decoding networks for correcting the "error" in the distributions between the data $\boldsymbol{x}$ and the decoded $\hat{\boldsymbol{x}}$. Using terminology from control theory, one may view the encoding network f as a "sensor" for error feedback while the decoding network g as a "controller" for error correction. However, notice that here the "target" for control is not a scalar nor a finite dimensional vector, but a continuous mapping—in order for the distribution of $\hat{\boldsymbol{x}}$ to match that of the data $\boldsymbol{x}$. This is in general a control problem in an infinite dimensional space. The space of diffeomorphisms of submanifolds is infinite-dimensional [1]. Ideally, we hope when the sensor f and the controller g are optimal, the distribution of $\boldsymbol{x}$ becomes a "fixed point" for the closed loop while the distribution of $\boldsymbol{z}$ reaches a compact LDR. Hence, the minimax programs (12) and (13) can also be interpreted as games between an error-feedback sensor and an error-reducing controller.

Remark: relation to bi-directional or cycle consistency. The notion of "bi-directional" and "cycle" consistency between encoding and decoding has been exploited in the works of BiGAN [38] and ALI [39] for mappings between the data and features and in the work of CycleGAN [56] for mappings between two different data distributions. In our context, it is similar in order to promote $g \circ f$ and $f \circ g$ to be close to identity mappings (either for the distributions or for the samples). Interestingly, our new closed-loop formulation actually "decouples" the data $\boldsymbol{X}$, say, observed from the external world, from their internally represented features $\boldsymbol{Z}$. The objectives (12) and (13) are functions of *only* the internal features $\boldsymbol{Z}(\theta)$ and $\hat{\boldsymbol{Z}}(\theta, \eta)$, which can be learned and optimized by adjusting the neural networks $f(\cdot, \theta)$ and $g(\cdot, \eta)$ alone. There is no need for any additional external metrics or heuristics to promote how "close" the decoded images $\hat{\boldsymbol{X}}$ are to $\boldsymbol{X}$. This is very different from most VAE/GAN-type methods such as BiGAN and ALI that require additional discriminators (networks) for the images and the features. Some experimental comparison are given in the Appendix A.2. In addition, in Appendix A.8.1, we provide some ablation study to illustrate the importance and benefit of a closed loop for enforcing the consistency between the encoder and decoder.

Remark: transparent versus hidden distribution of the learned features. Notice that in our framework, there is no need to explicitly specify a prior distribution either as a target distribution to map to for AE (2) or as an initial distribution to sample from for GAN (3). The common practice in AEs or GANs is to specify the prior distribution as a generic Gaussian. This is however particularly problematic when the data distribution is multi-modal and has multiple low-dimensional structures, which is commonplace for multi-class data. In this case, the common practice in AEs or GANs is to train a conditional GAN for different classes or different attributes. However, here we only need to assume

the desired target distribution belonging to the family of LDRs. The specific optimal distribution of the features within this family is then learned from the data directly, and then can be represented *explicitly* as a mixture of independent subspace Gaussians (or equivalently, a mixture of PCAs on independent subspaces). We will give more details in the experimental Section 3 as well as more examples in Appendices A.2–A.4. Although many GAN + VAE-type methods can learn bidirectional encoding and decoding mappings, the distribution of the learned features inside the feature space remains *hidden* or even *entangled*. This makes it difficult to sample the feature space for generative purposes or to use the features for discriminative tasks. (For instance, typically one can only use so-learned features for nearest-neighbor-type classifiers [38], instead of nearest subspace as in this work, see Section 3.3).

3. Empirical Verification on Real-World Imagery Datasets

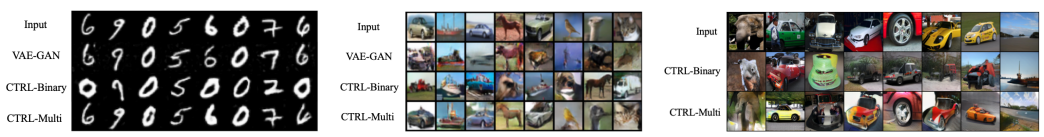
This experiment section serves three purposes: First, we empirically justify the proposed formulation for data transcription by demonstrating good properties of the learned encoder, decoder, and representation tuple (f, g, z) from $\boldsymbol{X}$. Second, we compare our method with several representative methods from the GAN family and VAE family. The purpose of the comparison is *not* to compete for any state-of-the-art performance. Instead, we want to convincingly verify the validity of the proposed framework and its potential in going beyond. Finally, we evaluate the so-learned CTRL through both generative tasks (controlled visualization) and discriminative (classification) tasks. More extensive experimental results, evaluations, and ablation studies can be found in the Appendix A.

Datasets. We provide extensive qualitative and quantitative experimental results on the following datasets: MNIST [57], CIFAR-10 [58], STL-10 [59], CelebA [60], LSUN bedroom [61], and ImageNet ILSVRC 2012 [62]. The network architectures and implementation details can be found in Appendix A.1 and corresponding Appendix A for each dataset.

3.1. Empirical Justification of CTRL Transcription

To empirically validate our new framework, we conduct experiments from a small low-variety dataset (MNIST), to a small dataset of diverse real-world objects (CIFAR-10), to higher resolution images (STL-10, CelebA, LSUN-bedroom), to a large-scale diverse image set (ImageNet). The results are evaluated both quantitatively and qualitatively. Implementation details, more experimental results, and ablation studies are given in Appendix A.

Comparison (IS and FID) with other formulations. First, we conduct five experiments to fairly compare our formulation with GAN [63] and VAE(-GAN) [64] on MNIST and CIFAR-10. Except for the objective function, everything else is exactly the same for all methods (e.g., networks, training data, optimization method). These experiments are: (1). GAN; (2). GAN with its objective replaced by that of the CTRL-Binary (13); (3). VAE-GAN ; (4). Binary CTRL (13); and (5). Multi-class CTRL (12). Some visual comparison is given in Figure 3. IS [65] and FID [66] scores are summarized in Table 1. Here, for simplicity, we have chosen a uniform feature dimension $d = 128$ for all datasets. If we choose a higher feature dimension, say $d = 512$, for the more complex CIFAR-10 dataset, the visual quality can be further improved, see Table A14 in Appendix A.11.



(**a**) MNIST (**b**) CIFAR-10 (**c**) ImageNet

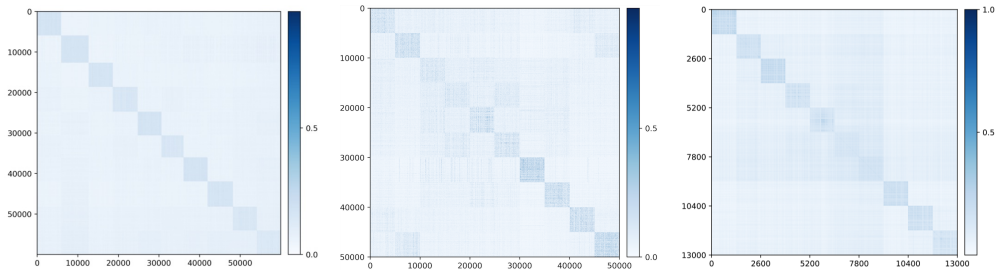
Figure 3. Qualitative comparison on (**a**) MNIST, (**b**) CIFAR-10 and (**c**) ImageNet. First row: original $\boldsymbol{X}$; other rows: reconstructed $\hat{\boldsymbol{X}}$ for different methods.

Table 1. Quantitative comparison on MNIST and CIFAR-10. Average Inception scores (IS) [65] and FID scores [66]. ↑ means higher is better. ↓ means lower is better.

Method		GAN	GAN (CTRL-Binary)	VAE-GAN	CTRL-Binary	CTRL-Multi
MNIST	IS ↑	2.08	1.95	**2.21**	2.02	2.07
	FID ↓	24.78	20.15	33.65	**16.43**	16.47
CIFAR-10	IS ↑	7.32	7.23	7.11	**8.11**	7.13
	FID ↓	26.06	22.16	43.25	**19.63**	23.91

As we see from Table 1, replacing cross-entropy with the Equation (13) can improve the generative quality. The two CTRL formulations are clearly on par with the others in terms of IS and significantly better in FID. Finally, with the same training datasets, the quality of CTRL-Multi is lower than that of CTRL-Binary. This is expected, as the multi-class task is more challenging. Nevertheless, as we will see soon, images decoded by CTRL-Multi align much better with their classes than Binary.

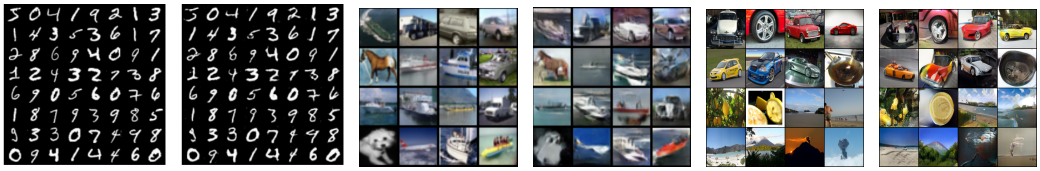
Visualizing correlation of features $\boldsymbol{Z}$ and decoded features $\hat{\boldsymbol{Z}}$. We visualize the cosine similarity between $\boldsymbol{Z}$ and $\hat{\boldsymbol{Z}}$ learned from the multi-class objective (12) on MNIST, CIFAR-10 and ImageNet (10 classes), which indicates how close $\hat{\boldsymbol{z}} = f \circ g(\boldsymbol{z})$ is from $\boldsymbol{z}$. Results in Figure 4 show that $\boldsymbol{Z}$ and $\hat{\boldsymbol{Z}}$ are aligned very well within each class. The block-diagonal patterns for MNIST are sharper than those for CIFAR-10 and ImageNet, as images in CIFAR-10 and ImageNet have more diverse visual appearances.



(**a**) MNIST (**b**) CIFAR10 (**c**) ImageNet

Figure 4. Visualizing the alignment between $\boldsymbol{Z}$ and $\hat{\boldsymbol{Z}}$: $|\boldsymbol{Z}^\top \hat{\boldsymbol{Z}}|$ and in the feature space for (**a**) MNIST, (**b**) CIFAR-10, and (**c**) ImageNet-10-Class.

Visualizing auto-encoding of the data $\boldsymbol{X}$ and the decoded $\hat{\boldsymbol{X}}$. We compare some representative $\boldsymbol{X}$ and $\hat{\boldsymbol{X}}$ on MNIST, CIFAR-10 and ImageNet (10 classes) to verify how close $\hat{\boldsymbol{x}} = g \circ f(\boldsymbol{x})$ is to $\boldsymbol{x}$. The results are shown in Figure 5, and visualizations are created from training samples. Visually, the auto-encoded $\hat{\boldsymbol{x}}$ faithfully captures major visual features from its respective training sample $\boldsymbol{x}$, especially the pose, shape, and layout. For the simpler dataset such as MNIST, auto-encoded images are almost identical to the original. The visual quality is clearly better than other GAN+VAE-type methods, such as VAE-GAN [34] and BiGAN [38]. We refer the reader to Appendices A.2, A.4 and A.7 for more visualization of results on these datasets, including similar results on transformed MNIST digits. More visualization results for learned models on real-life image datasets such as STL-10, CelebB, and LSUN can be found in the Appendices A.5 and A.6.



(**a**) MNIST $\boldsymbol{X}$ (**b**) MNIST $\hat{\boldsymbol{X}}$ (**c**) CIFAR-10 $\boldsymbol{X}$ (**d**) CIFAR-10 $\hat{\boldsymbol{X}}$ (**e**) ImageNet $\boldsymbol{X}$ (**f**) ImageNet $\hat{\boldsymbol{X}}$

Figure 5. Visualizing the auto-encoding property of the learned closed-loop transcription ($\boldsymbol{x} \approx \hat{\boldsymbol{x}} = g \circ f(\boldsymbol{x})$) on MNIST, CIFAR-10, and ImageNet (zoom in for better visualization).

3.2. Comparison to Existing Generative Methods

Table 2 gives a quantitative comparison of visual quality of our method with others on CIFAR-10, STL-10, and ImageNet. In general, there is a large difference in terms of FID and IS scores between the GAN family and the VAE family of models. SNGAN [31] are commonly used methods in most generative applications, while LOGAN [29] is the state-of-the-art method on ImageNet in terms of FID and IS. More comparisons with existing methods, including results on on the higher-resolution ImageNet dataset, can be found in Table A10 of the Appendix A.7.

As we see, even if the rate-reduction objectives (12) and (13) are not specifically designed nor engineered for visual quality and the networks and hyper-parameters adopted in our experiments are rather basic compared to many of the state-of-the-art generative methods, our method is still rather competitive in terms of these metrics. In our current implementation, the original objectives are used without any other heuristics or regularization. The simplicity of our framework and formulation suggests that there is significant room for further improvement. For instance, in all experiments on all datasets, we have chosen a feature dimension of $d = 128$ for simplicity and uniformity. In the last Appendix A.11, we have conducted an ablation study on using a higher feature dimension $d = 512$. The visual quality of the learned model can be significantly improved (as shown in Figure A22 and Table A14 of Appendix A.11).

In fact, compared to these methods, our method has learned not just any generative model. It has learned a *structured* generative model that has many additional beneficial properties that we now present.

Table 2. Comparison of CIFAR-10 and STL-10. Comparison with more existing methods and on ImageNet can be found in Table A10 in the Appendix A. ↑ means higher is better. ↓ means lower is better.

Method		**GAN Based Methods**			**VAE/GAN-Based Methods**				
		SNGAN	**CSGAN**	**LOGAN**	**VAE-GAN**	**NVAE**	**DC-VAE**	**CTRL-Binary**	**CTRL-Multi**
CIFAR-10	IS ↑	7.4	8.1	**8.7**	7.4	-	**8.2**	**8.1**	7.1
	FID ↓	29.3	19.6	**17.7**	39.8	50.8	**17.9**	**19.6**	23.9
STL-10	IS ↑	**9.1**	-	-	-	-	8.1	8.4	7.7
	FID ↓	40.1	-	-	-	-	41.9	**38.6**	45.7

3.3. Benefits of the Learned LDR Transcription Model

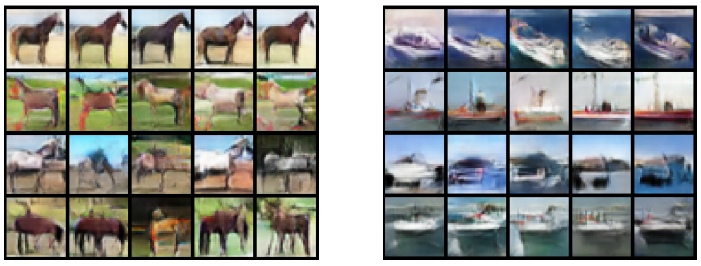
As we have argued before, the learned LDR transcription model (including the feature $\boldsymbol{z}$, the encoder f, and the decoder g) can be used for both generative and discriminative purposes. In particular, unlike almost all existing generative methods, the internal structures or distribution of the learned z are no longer "hidden" as they have clear subspace structures. Hence, we can easily derive an explicit (parametrizable) model for the distribution of the learned features as a mixture of independent subspace-like Gaussians. This gives us full control in sampling the learned distribution for generative purposes.

Principal subspaces and principal components for the feature. To be more specific, given the learned k-class features $\cup_{j=1}^{k} \boldsymbol{Z}_j$ for the training data, we have observed that the leading singular subspaces for different classes are all approximately orthogonal to each other: $\boldsymbol{Z}_i \perp \boldsymbol{Z}_j$ (see Figure 4). This corroborates with our above discussion about the theoretical properties of the rate-reduction objective. They essentially span k independent principal subspaces. We can further calculate the mean $\bar{\boldsymbol{z}}_j$ and the singular vectors $\{\boldsymbol{v}_j^i\}_{i=1}^{r_j}$ (or principal components) of the learned features $\boldsymbol{Z}_j$ for each class. Although we conceptually view the support of each class is a subspace, the actual support of the features is close to being on the sphere due to feature (scale) normalization. Hence, it is more precise to find its mean and its support centered around the mean. Here, r_j is a rank we may choose to model the dimension of each principal subspace (say, based on a common threshold on the singular values). Hence, we obtain an explicit model for how the feature $\boldsymbol{z}$ is distributed in each of the k principal subspaces in the feature space $\mathbb{R}^d$:

$$\boldsymbol{z}_j \sim \bar{\boldsymbol{z}}_j + \sum_{l=1}^{r_j} n_l^j \sigma_j^l \boldsymbol{v}_j^l, \quad \text{where} \quad n_l^j \sim \mathcal{N}(0,1), \;\; j = 1, \ldots, k. \tag{14}$$

Hence, this essentially gives an explicit mixture of a subspace-like Gaussians model for the learned features: statistical differences between different classes are modeled as k independent principal subspaces; statistical differences within each class j are modeled as r_j independent principal components in the jth subspace.

Decoding samples from the feature distribution. Using the CIFAR-10 and CelebA datatsets, we visualize images decoded from samples of learned feature subspace. For the CIFAR-10 dataset, for each class j, we first compute the top four principal components of the learned features $\boldsymbol{Z}_j$ (via SVD). For each class j, we then compute $|\langle \boldsymbol{z}_j^i, \boldsymbol{v}_j^l \rangle|$, the cosine similarity between the l-th principal direction $\boldsymbol{v}_j^l$ and feature sample $\boldsymbol{z}_j^i$. After finding the top five $\boldsymbol{z}_j^i$ according to $|\langle \boldsymbol{z}_j^i, \boldsymbol{v}_j^l \rangle|$ for each class j, we reconstruct images $\hat{\boldsymbol{x}}_j^i = g(\boldsymbol{z}_j^i)$. Each row of Figure 6 is for one principal component. We observe that images in the same row share the same visual attributes; images in different rows differ significantly in visual characteristics such as shape, background, and style. See Figure A7 of Appendix A.4 for more visualization of principal components learned for all 10 classes of CIFAR-10. These results clearly demonstrate that the principal components in each subspace of the Gaussian disentangles different visual attributes. In addition, we do not observe any mode dropping for any of the classes, although the dimensions of the classes were not known a priori.



(**a**) Horse (**b**) Ship

Figure 6. CIFAR-10 dataset. Visualization of top 5 reconstructed $\hat{\boldsymbol{x}} = g(\boldsymbol{z})$ based on the closest distance of $\boldsymbol{z}$ to each row (top 4) of principal components of data representations for class 7—'Horse' and class 8—'Ship'.

Disentangled visual attributes as principal components. For the CelebA dataset, we calculate the principal components of all learned features in the latent space. Figure 7a shows some decoded images along these principal directions. Again, these principal components seem to clearly *disentangle* visual attributes/factors such as wearing a hat, changing

hair color, and wearing glasses. More examples can be found in Appendix A.6. The results are consistent with *the property of MCR*^{2} *that promotes diversity of the learned features*.

(**a**) Disentangled attributes as principal components (**b**) Interpolation between distinct samples

Figure 7. CelebA dataset. (**a**): Sampling along three principal components that seem to correspond to different visual attributes; (**b**): Samples decoded by interpolating along the line between features of two distinct samples.

Linear interpolation between features of two distinct samples. Figure 7b shows interpolating features between pairs of training image samples of the CeleA dataset, where for two training images $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, we reconstruct based on their linearly interpolated feature representations by $\hat{\boldsymbol{x}} = g(\alpha f(\boldsymbol{x}_1) + (1-\alpha) f(\boldsymbol{x}_2)), \alpha \in [0,1]$. The decoded images show continuous morphing from one sample to another in terms of visual characteristics, as opposed to merely a superposition of the two images. Similar interpolation results between two digits in the MNIST dataset can be found in Figure A3 of the Appendix A.2.

Encoded features for classification. Notice that not only is the learned decoder good for generative purposes, but the encoder is also good for discriminative tasks. In this experiment, we evaluate the discriminativeness of the learned CTRL model by testing how well the encoded features can help classify the images. We use features of the training images to compute the learned subspaces for all classes, then classify features of the test images based on a simple nearest subspace classifier. Many other encoding methods train a classifier (say, with an additional layer) after the learned features. Results in Table 3 show that our model gives competitive classification accuracy on MNIST compared to some of best VAE-based methods. We also tested the classification on CIFAR-10, and the accuracy is currently about 80.7%. As expected, the representation learned with the multi-class objective is very discriminative and good for classification tasks. Be aware that all generative models, GANs, VAEs, and ours, are not specifically engineered for classification tasks. Hence, one should not expect the classification accuracy to compete with supervised-trained classifiers yet. This demonstrates that the learned CTRL model is not only generative but also discriminative.

Table 3. Classification accuracy on MNIST compared to classifier-based VAE methods [42]. Most of these VAE-based methods require auxiliary classifiers to boost classification performance.

Method	VAE	Factor VAE	Guide-VAE	DC-VAE	CTRL-Binary	CTRL-Multi
MNIST	97.12%	93.65%	98.51%	98.71%	89.12%	98.30%

4. Open Theoretical Problems

So far, we have given theoretical intuition and derivation for the formulation of closed-loop transcription, as well as empirical evidence to showcase both the performance and potential of this formulation. In this section, we take a step back to explore the theoretical underpinnings of the closed-loop LDR transcription. We organize this section by discussing three primary objectives associated with learning an LDR representation:

1. *Learn a simple linear discriminative representation* $f(\boldsymbol{X})$ *of the data* $\boldsymbol{X}$, which we can reliably use to classify the data.

2. *Learn a reconstruction* $g \circ f(\boldsymbol{X}) \sim \boldsymbol{X}$ *of the so-learned representation* $f(\boldsymbol{X})$, to ensure consistency in the representation.
3. *Learn both representation and reconstruction in a closed-loop manner*, using feedback from the encoder f and decoder g to jointly solve the above two tasks.

These three objectives encompass the overarching principle of CTRL transcription, and indeed each of these objectives are tied to a wide array of mathematical and theoretical problems. We now outline some of the most important theoretical questions or hypotheses implicated by our results, which we leave for future work to study and to answer, likely by a broader range of research communities.

4.1. Distributions of the LDR Representation

Our primary mode of optimizing for a "simple representation" is through the LDR framework proposed in [2]. One important open theoretical problem is finding the right energy function to optimize in order to promote LDR. It was shown in [2] that an LDR can be learned for the multi-class data by maximizing the MCR^2 objective $\Delta R(\boldsymbol{Z})$ given in (5). This motivates the first two terms in our objective function (12): maximizing $\Delta R(\boldsymbol{Z}), \Delta R(\hat{\boldsymbol{Z}})$ promotes their representations to be LDRs.

Although the authors in [2] have shown the MCR^2 objective can promote the features learned to be in orthogonal subspaces and characterized the optimal second moments of the distributions, there remain open questions regarding the optimal distributions within the subspaces. A standing hypothesis is that the optimal distributions should be Gaussian. There is indeed already theoretical work on similar energy functions: the Brascamp–Lieb inequalities [67], where the authors study a functional similar to the rate-reduction objective which, in certain contexts, is maximized uniquely by Gaussians. Hence, an important future theoretical direction for the CTRL transcription is to exactly characterize distributional properties of the extremals (both minima and maxima) of the MCR^2 objective or its variants. Such results can further justify the use of Gaussian models (14) to characterize the learned features within the subspaces.

We also notice that the so-learned LDR features have additional striking properties, as shown by examples in Figure 7. Distinctive visual attributes of the imagery data seem to be clearly disentangled by different principal components of the distribution, and along each principal direction, one can linearly interpolate the features, whereas the original data are nonlinear and cannot be directly interpolated. These results go beyond the guarantees given by [2], and an open theoretical problem is that of studying just how the CTRL transcription learns to disentangle and linearize such visual attributes. This understanding is crucial to extend the CTRL transcription framework beyond the 2D vision domain.

4.2. Self-Consistency in the Learned Reconstruction

If the learned encoder $\boldsymbol{Z} = f(\boldsymbol{X})$ is an embedding of the data submanifolds to the subspaces, it should admit an inverse (decoding) mapping $\hat{\boldsymbol{X}} = g(\boldsymbol{Z})$. As distributional distance in the data space is hard to come by, the rate reduction $\Delta R(\boldsymbol{Z}, \hat{\boldsymbol{Z}})$ gives a well-defined distribution distance between $\boldsymbol{Z}$ and $\hat{\boldsymbol{Z}}$ which is used to enforce similarity between $\boldsymbol{X}$ and $\hat{\boldsymbol{X}}$ in our formulation. Notice that, unlike the KL-divergence or the JS-divergence, the rate reduction is well-defined for degenerate distributions and easily computable in closed-form between mixtures of (degenerate) Gaussians. The third term of Equation (12), $\sum_{j=1}^{k} \Delta R\big(\boldsymbol{Z}_j(\theta), \hat{\boldsymbol{Z}}_j(\theta, \eta)\big)$, is exactly this distributional distance, which is minimized only when the estimated second moments of $\boldsymbol{Z}_j$ and $\hat{\boldsymbol{Z}}_j$ are the same. While this distributional distance seems weaker than sample-wise ℓ^2-distance, we observe strong reconstruction performance nevertheless.

Notice that the current objectives (12) or (13) do not impose any constraints on the mappings of individual samples. That is, they do not explicitly specify how an individual sample $\boldsymbol{x}$ should be related to its decoded version $\hat{\boldsymbol{x}} = g(f(\boldsymbol{x}))$, or how their corresponding features $\boldsymbol{z}$ and $\hat{\boldsymbol{z}}$ are related. Hence, theoretically, nothing is known about relationships between individual samples and their features. However, somewhat surprisingly, experi-

mental results with the multi-class objective (12) in next section suggest that they actually can be rather close, at least for the given training samples $\boldsymbol{X}$. For example, see Figure 5. Of course, one could consider explicitly imposing certain sample-wise requirements in the objectives, such as enforcing $\boldsymbol{x}^i$ to be close to $\hat{\boldsymbol{x}}^i = g(f(\boldsymbol{x}^i))$. It has been observed empirically in GANs or VAEs that imposing such sample-wise similarity or dissimilarity would improve visual quality around samples of interest, such as the DC-VAE [42] and the OpenGAN [68]. However, theoretically, how such sample-wise distances or constraints may affect the difficulty or accuracy of learning the correct support and density of the distributions remains an open problem.

4.3. Properties of the Closed-Loop Minimax Game

Above are the two primary objectives for CTRL transcription: while the encoder f tries to maximize the expressiveness and discriminativeness of the learned LDR representation, the decoder g tries to minimize the reconstruction error and coding rates. The competing objectives of the encoder f and the decoder g naturally lead to a two-player game. In this paper, we have formulated this game as a zero-sum game, namely Equation (12). Likewise, we have also implemented the most straightforward algorithm for solving this zero-sum game: gradient descent–ascent (GDA) [52], where the minimizer and maximizer take alternating gradient steps. These simplifications into a GDA-optimized zero-sum game were made in order to create a concrete algorithm for our experimentation. However, simplifying to a zero-sum game and GDA is certainly not the only way to solve the more general game described above. This game-theoretic formulation puts CTRL transcription outside of the theoretical realm of [2], since we are no longer finding pure maximizers of $\Delta R(\boldsymbol{Z})$, but rather stable minimax equilibria.

As is the case with GANs, these equilibria may not necessarily be Nash equilibria [50], but rather the more general sense of Stackelberg [69]. So, the problem of studying minimax equilibria of (12) is likely, in its most general form, quite challenging. Nevertheless, our experiments suggest such equilibria tend to be well-behaved, e.g., having a large range of attraction. Our extensive empirical experiments and ablation studies indicate that, in general, the minimax objective converges rather stably to good equilibria for all the real datasets without any special optimization tricks or particular requirements on the networks. The only important factor for the stability of the optimization seems to be a large enough batch size (see Appendix A.10). These observations can be further corroborated with analysis on simpler models: our ongoing work suggests that if we restrict our attention to simplified data structures (e.g., $\boldsymbol{X}$ distributed on a linear subspace), then one can provide theoretical guarantees that the equilibria become efficiently and correctly solvable by the minimax formulation. Extending such analysis to more sophisticated data structures (multiple subspaces, nonlinear submanifolds) remains an exciting new directions for future research.

Despite many possible pathological solutions to the minimax game, empirically, as we have presented in the previous section (alongside many examples in the Appendix A), the solution found by the simple GDA algorithm generally strikes a good trade-off between expressiveness and parsimony of the learned model. The solution automatically determines the proper dimensions for different classes. Ablation studies in Appendix A.10 on the large ImageNet dataset further suggest that this formulation is insensitive to overparameterization by increasing network width, as long as the batch size grows accordingly. However, a rigorous justification for such good model-selection properties remains widely open.

5. Conclusions and Future Work

This work provides a novel formulation for learning a *both generative and discriminative* representation for a multi-class, multi-dimensional, possibly nonlinear, distribution of real-world data. We have provided compelling empirical evidence that the distribution of most datasets can be effectively mapped to an LDR, a union of independent princi-

pal subspaces and principal components. The objective function is entirely based on an intrinsic information-theoretic measure, the rate reduction, without any other heuristics or regularizing terms. The objective can be achieved with a closed-loop minimax game between the two encoder and the decoder networks without any additional network(s).

The main purpose of this paper is to demonstrate the conceptual simplicity and practical potential of this new framework for distribution/representation learning, instead of striving for state-of-the-art performance with heavy engineering. Nevertheless, with our preliminary implementation, a more informative LDR of the data can be effectively learned with a simple closed-loop transcription for a variety of real-world, multi-class, multi-modal visual datasets, from small to large, from low-resolution to higher-resolution, from domain-specific to diverse categories. The so-learned encoder f already enjoys the benefits of AE/VAEs for their discriminative property and the decoder g with the benefits of GANs for their good generative visual quality. However, probably more importantly, the internal structures of the learned feature representation has now become transparent, hence *fully interpretable and controllable* (for generative purposes): visual differences between classes are naturally "disentangled" as independent subspaces, while diverse visual attributes within each class are "disentangled" as principal components within each subspace. From extensive ablation studies given in the Appendix A, we see that the rate-reduction-based objective can be stably optimized across a wide range of datasets and network architectures without any additional regularizations or engineering tricks. Both the *feedback closed-loop* and the *rate-reduction measure* play indispensable roles in fostering the ease and success of finding the CTRL transcription.

One may notice that there are many ways this simple formulation can be significantly improved or extended. Firstly, in this work, we have simply adopted networks that were designed for GANs, but they may not be optimal for the rate-reduction-type objectives. For example, our ablation study already suggests that some of the components of such networks such as spectral normalization are not quite essential. Characteristics from the white-box ReduNet [2] derived from optimizing rate reduction can be explored in the future. Secondly, notice that our rate-reduction objectives do not impose any requirements on how individual samples should be encoded or decoded although the results from the multi-class objective indicate a certain level of alignment on the individual samples. Recent studies such as DC-VAE [42] or OpenGAN [68] suggest that imposing additional regularization on individual samples may further improve decoded visual quality. Such regularization can certainly be incorporated into this new framework. Last but not the least, compared to GANs and VAEs, our method leads to an *explicit* structured model for the feature distribution: a mixture of incoherent subspace Gaussians. Such an explicit model has the potential of making many subsequent tasks easier and better: better control of feature sampling for decoding and synthesis [70], designing more robust generators and classifiers for noise and corruptions based on the low-dimensional structures identified, or even extending to the settings of incremental and online learning [71,72]. We leave all these new directions, together with all the open theoretical problems posed in Section 4, for future investigation.

Author Contributions: This work has been the result of a successful team effort. In particular, the first four authors have contributed almost equally to this work. X.D.: investigation, methodology, project administration, software, writing—original draft preparation; S.T.: investigation, methodology, software, visualization, writing—original draft preparation; M.L.: investigation, software, visualization, writing—original draft preparation; Z.W.: investigation, software, visualization, writing—original draft preparation; M.P.: formal analysis, writing—original draft preparation; K.H.R.C.: validation, writing—review and editing; P.Z.: formal analysis, writing—review and editing; Y.Y.: validation, writing—review and editing; X.Y.: resources, writing—review and editing; H.-Y.S.: resources, writing—review and editing; Y.M.: conceptualization, formal analysis, funding acquisition, methodology, supervision, writing—original draft preparation, writing—review and editing; All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by ONR grants N00014-20-1-2002 and N00014-22-1-2102, the joint Simons Foundation-NSF DMS grant #2031899, as well as partial support from Berkeley FHL Vive Center for Enhanced Reality and Berkeley Center for Augmented Cognition, Tsinghua-Berkeley Shenzhen Institute (TBSI) Research Fund, and Berkeley AI Research (BAIR).

Data Availability Statement: Data and results can be found in Section 3 and Appendix A.

Acknowledgments: Earliest ideas of this work were germinated during a hiking event of Ma's group on Berkeley hills during the summer of 2020. Former group members Chong You (now at Google) and Yichao Zhou (now at Apple) were part of a stimulating discussion on possible extensions or applications of a new rate-reduction framework being developed then. During the preparation of this work, we consulted several experts on some of the related topics. The authors would like to thank Jiantao Jiao of UC Berkeley for discussions about the theoretical conditions for learning distributions via GANs. We thank Benjamin Haeffele of Johns Hopkins University for sharing thoughts on how to learn subspaces correctly and on how to optimize the rate-reduction objectives efficiently. We would also like to thank Shankar Sastry and Manxi Wu of UC Berkeley and Chaobing Song of Univ. of Wisconsin-Madison for informative discussions on how to solve minimax games correctly and efficiently, as well as Chih-Yuan Chiu and Druv Pai of UC Berkeley for engaging discussions on theoretical directions for the CTRL transcription. Last but not the least, we would like to thank Stefano Soatto of UCLA for stimulating discussions and sometimes heated debates on how information can be efficiently and effectively encoded in deep networks.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

Appendix A.1. Experiment Settings and Implementation Details

Network backbones. For MNIST, we use the standard CNN models in Tables A1 and A2, following the DCGAN architecture [63]. We resize the MNIST image resolution from 28×28 to 32×32 to fit DCGAN architecture. All α in lReLU (lReLU is short for Leaky-ReLU) of the encoder are set to 0.2.

We adopt ResNet architectures for CIFAR-10 shown in Tables A3 and A4, and STL-10 shown in Tables A5 and A6. Each ResBlock up is same as Resnet, but add an up-sampler after the first conv layer. All batch normalization layers of ResBlock in the encoder are replaced with spectral normalization layer.

Finally, we use the same architecture for CelebA, LSUN-bedroom, and ImageNet-128 (see Tables A7 and A8) as all three datasets have the same 128×128 resolution. Again, each ResBlock up is same as Resnet, but add an up-sampler after the first conv layer. All batch-normalization layers in the encoder are replaced with spectral normalization layer. All experiments utilize this lightweight PyTorch library "mimicry" [73] that provides implementations of some popular state-of-the-art GANs and evaluation metrics.

Table A1. Decoder for MNIST.

$z \in \mathbb{R}^{1\times1\times128}$
4×4, stride = 1, pad = 0 deconv. BN 256 ReLU
4×4, stride = 2, pad = 1 deconv. BN 128 ReLU
4×4, stride = 2, pad = 1 deconv. BN 64 ReLU
4×4, stride = 2, pad = 1 deconv. 1 Tanh

Table A2. Encoder for MNIST.

Gray image $\boldsymbol{x} \in \mathbb{R}^{32\times32\times1}$
4×4, stride = 2, pad = 1 conv 64 lReLU
4×4, stride = 2, pad = 1 conv. BN 128 lReLU
4×4, stride = 2, pad = 1 conv. BN 256 lReLU
4×4, stride = 1, pad = 0 conv 128

Table A3. Decoder for CIFAR-10.

$\boldsymbol{z} \in \mathbb{R}^{128}$
dense $\rightarrow 4 \times 4 \times 256$
ResBlock up 256
ResBlock up 256
ResBlock up 256
BN, ReLU, 3×3 conv, 3 Tanh

Table A4. Encoder for CIFAR-10.

RGB image $\boldsymbol{x} \in \mathbb{R}^{32\times32\times3}$
ResBlock down 128
ResBlock down 128
ResBlock 128
ResBlock 128
ReLU
Global sum pooling
dense $\rightarrow$ 128

Table A5. Decoder for STL-10.

$\boldsymbol{z} \in \mathbb{R}^{128}$
dense $\rightarrow 6 \times 6 \times 512$
ResBlock up 256
ResBlock up 128
ResBlock up 64
BN, ReLU, 3×3 conv, 3 Tanh

Table A6. Encoder for STL-10.

RGB image $x \in \mathbb{R}^{48\times48\times3}$
ResBlock down 64
ResBlock down 128
ResBlock down 256
ResBlock down 512
ResBlock 1024
ReLU
Global sum pooling
dense $\rightarrow$ 128

Table A7. Decoder for CelebA-128, LSUN-bedroom-128, and ImageNet-128.

$z \in \mathbb{R}^{128}$
dense $\rightarrow 4 \times 4 \times 1024$
ResBlock up 1024
ResBlock up 512
ResBlock up 256
ResBlock up 128
ResBlock up 64
BN, ReLU, 3×3 conv, 3 Tanh

Table A8. Encoder for CelebA-128, LSUN-bedroom-128, and ImageNet-128.

RGB image $x \in \mathbb{R}^{128\times128\times3}$
ResBlock down 64
ResBlock down 128
ResBlock down 256
ResBlock down 512
ResBlock down 1024
ResBlock 1024
ReLU
Global sum pooling
dense $\rightarrow$ 128

Optimization and training details. Across all of our experiments, we use Adam [74] as our optimizer, with hyperparameters $\beta_1 = 0.5, \beta_2 = 0.999$. We adopt the simple gradient descent–ascent algorithm for alternating minimizing and maximizing the objectives. The initial value of learning rate is set to be 0.00015 and is scheduled with linear decay. We choose $\epsilon^2 = 0.5$ for both Equations (12) and (13) in all CTRL experiments. For all CTRL-Multi experiments on ImageNet, we only choose 10 classes. The details of the 10 classes are shown in Table A9. Most experiments are trained on RTX 3090 GPUs.

Table A9. ID and correspond category for 10 classes of ImageNet.

ID	Category
n02930766	cab, hack, taxi, taxicab
n04596742	wok
n02974003	car wheel
n01491361	tiger shark, Galeocerdo cuvieri
n01514859	hen
n09472597	volcano
n07749582	lemon
n09428293	seashore, coast, seacoast, sea-coast
n02504458	African elephant, Loxodonta africana
n04285008	sports car, sport car

Appendix A.2. MNIST

Settings. On MNIST dataset, we train our model using DCGAN [63] architecture with our proposed objectives CTRL-Multi (12) and CTRL-Binary (13). The learning rate is set to 10^{-4} and the batch size is set to 2048. We train our model with 15,000 iterations.

More results illustrating auto-encoding. Here, we give more reconstruction results, or $\hat{\boldsymbol{X}}$, from CTRL-Multi and CTRL-Binary objectives, compared to their corresponding original input $\boldsymbol{X}$. As shown in the Figure A1, for the CTRL-Binary objective, it can generate clean digit-like images but the decoded $\hat{\boldsymbol{X}}$ might resemble digits from similar but different classes to the input data $\boldsymbol{X}$ since the CTRL-Binary tends to only align the distribution of all digits.

In contrast, with the CTRL-Multi objective, the decoded $\hat{\boldsymbol{X}}$ not only are coherent with the correct class with the input data $\boldsymbol{X}$, but also show very clear one-to-one mapping between individual samples $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$, although the objective (12) does not enforce that. Comparing with the results from VAE-GAN [34] and BiGAN [38], our decoded images make less errors in reconstruction and preserve much better the individual characteristics of the original samples.

(**a**) Original $\boldsymbol{X}$ (**b**) VAE-GAN $\hat{\boldsymbol{X}}$ (**c**) BiGAN $\hat{\boldsymbol{X}}$ (**d**) CTRL-Binary $\hat{\boldsymbol{X}}$ (**e**) CTRL-Multi $\hat{\boldsymbol{X}}$

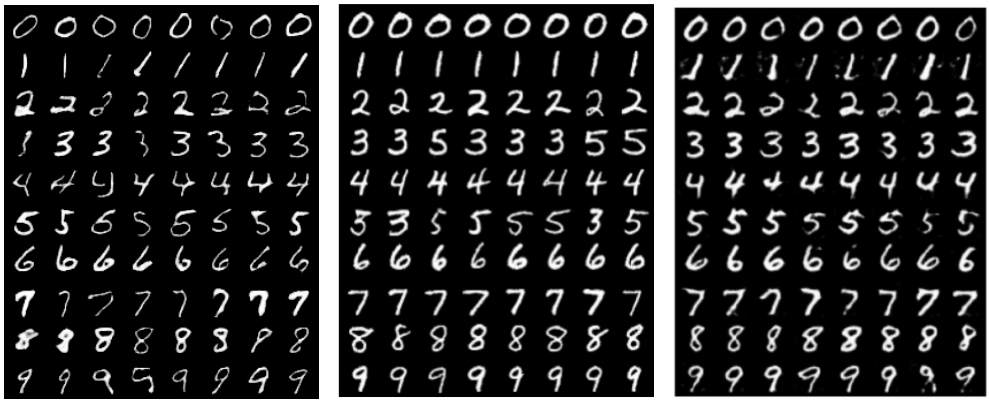
Figure A1. The comparison of the reconstruction results of different methods with the input data.

Images decoded from random samples on the learned multi-class LDR. Since our CTRL-Multi objective function maps input data of each class into a different (orthogonal) subspace in the feature space, we can generate images conditioned on each class by random sampling $\boldsymbol{z}$ in the subspace of each class and then decode them back to the input space as $\hat{\boldsymbol{x}}$.

To perform random sampling in the learned subspace, we first calculate the mean feature $\bar{\boldsymbol{z}}_j$ and the singular vectors $\boldsymbol{v}_j^i$ from the SVD (or principal components) of the learned features $\boldsymbol{Z}_j$ of the training data in the class j, where index i represents the ith principal components. We only use the top $r = 8$ principal components of each class on MNIST dataset. These statistics of the subspace can be used for guiding the random sampling. Then, we sample $\boldsymbol{z}$ randomly along the principal components and around the mean feature as

$$\boldsymbol{z}_{random\_j} = \bar{\boldsymbol{z}}_j + \alpha \sum_{i=1}^{r} n_i * \sigma_j^i * \boldsymbol{v}_j^i, \tag{A1}$$

where $\bar{z}_j$ is the mean feature of class j, σ_j^i and v_j^i are the i-th singular value and principal component of class j, n_i are i.i.d. Gaussian $\mathcal{N}(0,1)$ random variables. That is, the feature in each subspace/class is modeled by an r-dimensional multivariate Gaussian, with variances σ_j^i which characterize variances of the training data in the feature space. Here, α is a hyper-parameter that controls the sampling range. As for the visualization of random generated images $g(z_{random\_j})$ conditioned on the given class, we compare our method with some other conditional generation methods such as ACGAN [25] and InfoGAN [21] (for ACGAN and InfoGAN, we generate images conditioned on class labels with randomly sampled latent z according the procedures mentioned in their respective works). Our model can give realistic and correct conditional generation results with high diversity in each class, while other methods may make mistakes in the generation between some similar classes such as classes 3 and 5 for InfoGAN.



(**a**) ACGAN (**b**) InfoGAN (**c**) CTRL-Multi

Figure A2. Comparison of randomly generated images conditioned on each class.

Interpolation between samples in different classes. We randomly sample some images from each class. For each image x_1, we randomly sample another image x_2 from a different class. For such a pair of images x_1 and x_2, we reconstruct them based on their linearly interpolated feature representations by $\hat{x} = g(\alpha f(x_1) + (1-\alpha)f(x_2)), \alpha \in [0,1]$, the results of which are shown in the Figure A3. For each row in the figure from left to the right, the reconstructed images continuously morph from one digit to a different digit with a natural transition in shape rather than a simple superposition of the two images. This also confirms that space between subspaces for the digits does not represent valid digits but only shapes with digit-like strokes. Hence for generative purposes, knowing the supports of valid digits is extremely important.

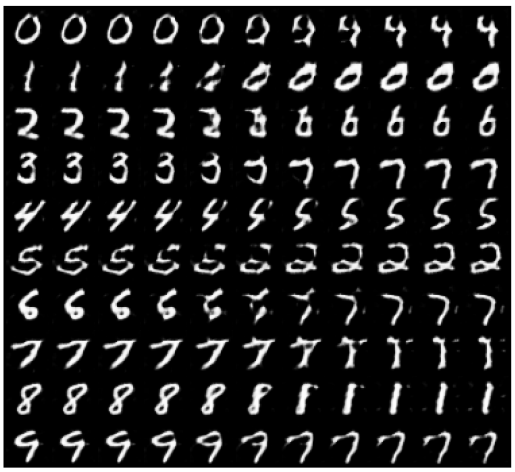
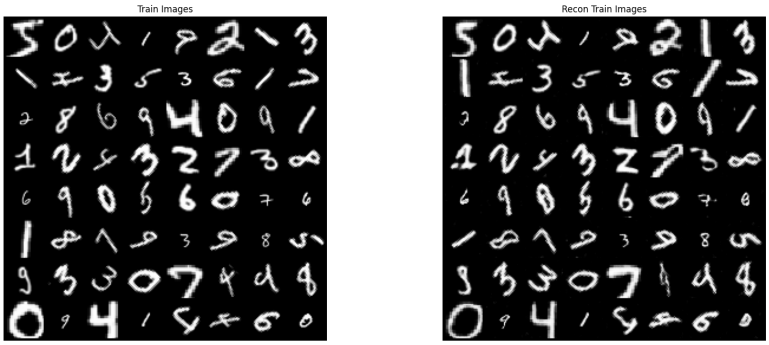


Figure A3. Images generated from the interpolation between samples in different classes.

Appendix A.3. Transformed MNIST

Settings. In this experiment, we verify that the CTRL-Multi objective can preserve diverse data modes in the learned feature embeddings. We construct a transformed MNIST dataset with five modes: normal, large (1.5×), small (0.5×), rotate 45° left, and rotate 45° right. Each image data point will be randomly transformed to one of the modes. Representative examples of such training data can be found in Figure A4a. We train the model with learning rate 1×10^{-4} and batch size 2048 for 15,000 iterations.

Auto-encoding results. Figure A4b gives the decoded results of the training data with different modes. Even though the data are now much more diverse for each class, decoder learned from the CTRL-Multi objective can still achieve high sample-wise similarity to the original images.

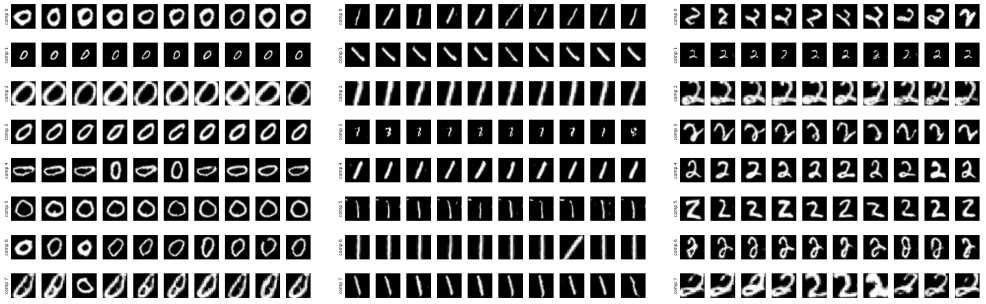


(**a**) Original $\boldsymbol{X}$ (**b**) Decoded $\hat{\boldsymbol{X}}$

Figure A4. Original (training) data $\boldsymbol{X}$ and their decoded version $\hat{\boldsymbol{X}}$ on the transformed MNIST.

Identifying different modes. Similar to the earlier experiments of Figure 6 for CIFAR-10 in the main paper, we find the top principal components of features of each class $\boldsymbol{Z}_j$ (via SVD) and generate new images using the learned decoder g from features of the training images aligned the best with these components.

In Figure A5, we select three classes 0, 1, 2 and visualize samples from the top $r = 8$ principal components for each class. Each row represents one principal component direction. As can be seen in the figure, the decoded images along each principal component shows a similar mode and the modes along different component directions are rather incoherent. All major modes of the original data can be identified as one of these principal component directions. This clearly shows that the CTRL-Multi objective can keep the different modes within each class of the data $\boldsymbol{X}_j$ as the principal component directions of $\boldsymbol{Z}_j$, and these modes can also be retained in the decoded images $\hat{\boldsymbol{X}}_j$.



(**a**) Components of class "0" (**b**) Components of class "1" (**c**) Components of class "2"

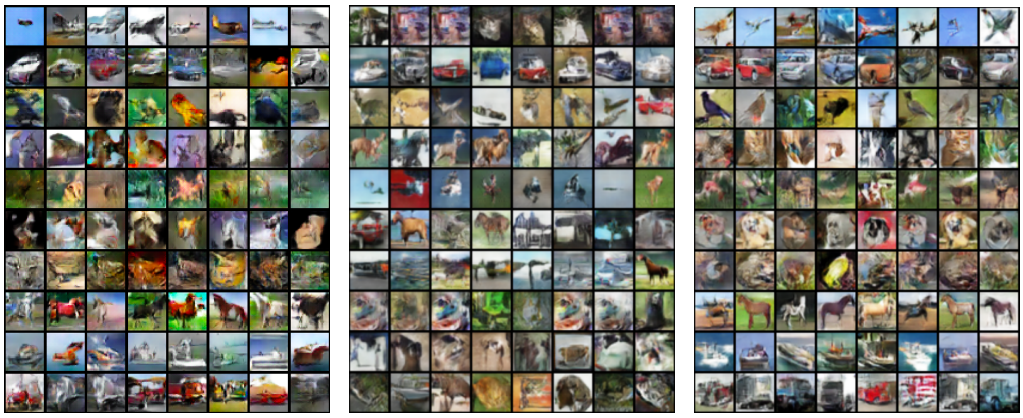
Figure A5. The reconstructed images $\hat{\boldsymbol{X}}$ from the features $\boldsymbol{Z}$ best aligned along top-8 principal components on the transformed MNIST dataset. Each row represents a different principal component.

Appendix A.4. CIFAR-10

Settings. For all experiments on CIFAR-10, we follow the common training hyper-parameters in Appendix A.1. Beyond that, for each experiment, we run 450,000 iterations with batch size 1600.

Images decoded from random samples on the CTRL-Multi. We sample z in the feature space randomly along the principal components and around the mean feature of each class $\boldsymbol{Z}_j$ as in the MNIST case, according to Equation (A1). The generated images from the sampled features are illustrated in Figure A6, one row per class. As we see, the generator learned from the CTRL-Multi objective is capable of generating diverse images for each class.

Further, for visualization of random generated images $g(\boldsymbol{z}_{random\_j})$ conditioned on the given class, we compare our method with some other conditional generation method such as ACGAN [25] and InfoGAN [21]. For all three experiments, we have randomly sampled 8 images per class in CIFAR-10. For more complex datasets such as CIFAR-10, our model can give more realistic conditional generation results for different classes with high diversity within each class.



(**a**) ACGAN (**b**) InfoGAN (**c**) CTRL-Multi

Figure A6. Comparison of randomly generated images conditioned on each class.

Generating images along different PCA components for each class. For each class, we first compute the top 10 principal components (singular vectors of the SVD) of $\boldsymbol{Z}$ and then for each of the top singular vectors, we display in each row the top 10 reconstructed image $\hat{\boldsymbol{X}}$ whose $\boldsymbol{Z}$ are closest to the singular vector using methods described in the main body of the paper, Section 3.3. The results are given in Figure A7. Notice that images in each row are very similar as they are sampled along the same principal component, whereas images in different rows are very different as they are orthogonal in the feature space. These results indicate that the features learned by our method can not only disentangle different classes as orthogonal subspaces but can also disentangle different visual attributes within each class as (orthogonal) principal components within each subspace.

(**a**) Airplane (**b**) Automobile (**c**) Bird

(**d**) Cat (**e**) Deer (**f**) Dog

(**g**) Frog (**h**) Horse (**i**) Ship

(**j**) Truck

Figure A7. Reconstructed images $\hat{X}$ from features Z close to the principal components learned for the 10 classes of CIFAR-10.

Appendix A.5. STL-10

Settings. For all experiments on STL-10, we follow the common training hyper-parameters in Appendix A.1. For the CTRL-Binary setting, we train 150,000 iterations. For the CTRL-Multi setting, we initialize the weights from the 20,000-th iteration of CTRL-

Binary checkpoint and train for another 80,000 iterations (with the CTRL-Multi objective). The IS and FID scores on the STL-10 dataset are reported in Table A10, on par or even better than existing methods such as SNGAN [31] or DC-VAE [42].

Visualizing auto-encoding property for the CTRL-Binary. We visualize the original images $\boldsymbol{x}$ and their decoded $\hat{\boldsymbol{x}}$ generated by the LDR model learned from the CTRL-Binary objective. The results are shown in Figure A8 for STL-10.

(**a**) Original $\boldsymbol{X}$ (**b**) Decoded $\hat{\boldsymbol{X}}$

Figure A8. Visualizing the original $\boldsymbol{x}$ and corresponding decoded $\hat{\boldsymbol{x}}$ results on STL-10 dataset. Note the model is trained from the CTRL-Binary objective hence sample- or class-wise correspondence is relatively poor, but the decoded image quality is very good.

Appendix A.6. Celeb-A and LSUN

To verify that our formulation works on images of higher resolution, we conduct experiments on the Celeb-A and LSUN datasets, which have a resolution of 128×128.

Settings. For all experiments on these datasets, we follow the common training hyper-parameters in Appendix A.1. We choose a 300 batch size for Celeb-A and LSUN. Both of them are trained with the CTRL-Binary objective and for 450,000 iterations.

Generating images along different PCA components. We calculate the principal components of the learned features $\boldsymbol{Z}$ in the latent subspace. We manually choose three principle components which are related to hat, hair color, and glasses (see Figure A9). The three components are 9th, 19th, and 23rd respectively from the overall 128 principal components. These principal directions seem to clearly disentangle visual attributes/factors such as wearing a hat, changing hair color, and wearing glasses.

Images generated from random sampling of the feature space. We sample z randomly according to the following Gaussian model:

$$\boldsymbol{z}_{random} = \bar{\boldsymbol{z}} + \alpha \sum_{i=1}^{r} n_i * \sigma_i * \boldsymbol{v}_i, \tag{A2}$$

where $\bar{\boldsymbol{z}}$ is the mean feature, σ_i and $\boldsymbol{v}_i$ are the ith singular value and singular vector, respectively, n_i are i.i.d. Gaussian $\mathcal{N}(0,1)$ random variables. As before α is a hyper-parameter to control the sampling range. We use the top r = 100 principle components for random sampling. The random generated images are realistic and diverse (see Figure A10).

Visualizing auto-encoding property for CTRL-Binary. We visualize the original image $\boldsymbol{x}$ and their decoded $\hat{\boldsymbol{x}}$ using the LDR model learned from the CTRL-Binary objective. The results are shown in Figures A11 and A12 for the Celeb-A dataset and the LSUN dataset, respectively. The CTRL-Binary objective can give very good visual quality for $\hat{\boldsymbol{x}}$ but cannot ensure sample-to-sample alignment. Nevertheless, the decoded $\hat{\boldsymbol{x}}$ seems to be very similar to the original $\boldsymbol{x}$ in some main visual attributes. We believe the binary objective manages to align only the dominant principal component(s) associated with the most salient visual attributes, say, pose of the face for Celeb-A or layout of the room for LSUN, between features of $\boldsymbol{X}$ and $\hat{\boldsymbol{X}}$.

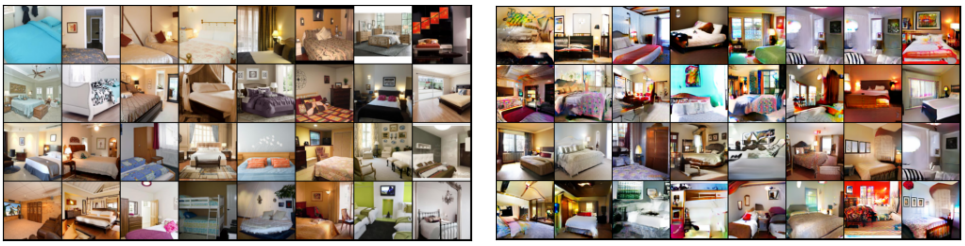
(**a**) Hat (**b**) Hair Color (**c**) Glasses

Figure A9. Sampling along the 9th, 19th, and 23rd principal components of the learned features $\boldsymbol{Z}$ seems to manipulate the visual attributes for generated images on the CelebA dataset.

Figure A10. Images decoded from randomly sampled features, as a learned Gaussian distribution (A2), for the CelebA dataset.

(**a**) Original $\boldsymbol{X}$ (**b**) Decoded $\hat{\boldsymbol{X}}$

Figure A11. Visualizing the original x and corresponding decoded $\hat{x}$ results on Celeb-A dataset. The LDR model is trained from the CTRL-Binary objective.



(**a**) Original $\boldsymbol{X}$ (**b**) Decoded $\hat{\boldsymbol{X}}$

Figure A12. Visualizing the original $\boldsymbol{x}$ and corresponding decoded $\hat{\boldsymbol{x}}$ results on LSUN-bedroom dataset. The LDR model is trained from the CTRL-Binary objective.

Appendix A.7. ImageNet

Settings. To verify that the CTRL works on large-scale datasets, we train it on the ImageNet. For all experiments on the ImageNet, we follow the common training hyper-parameters in Appendix A.1.

We first train our model with the CTRL-Binary objective with batch size of 1800 on the whole ImageNet ILSVRC 2012 dataset. The number of training iterations is 450,000.

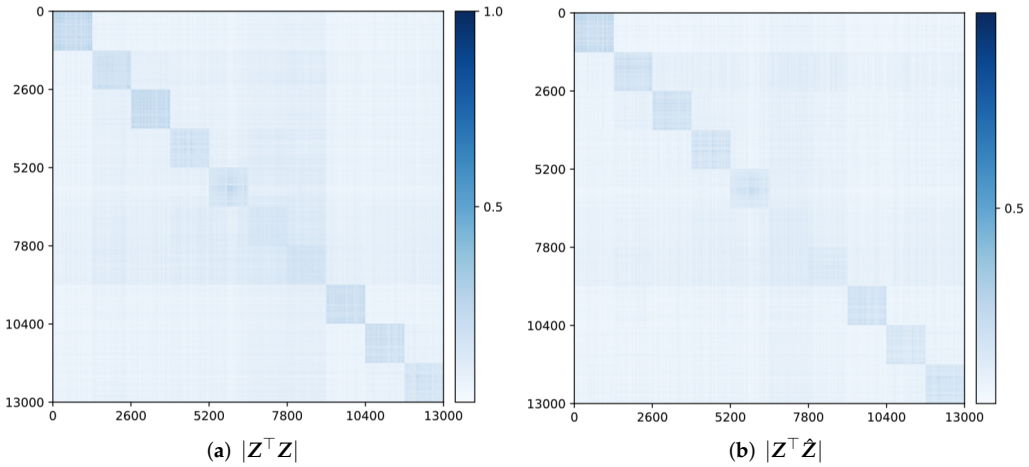
After that, we fine-tune the pretrained model with the CTRL-Multi objective, on 10 selected classes. Information about the 10 classes can be found in Table A9. The fine-tune batch size is 1024, and we train another 35,000 iterations for it. This experiment takes 120 GPU hours on 8 A100-SXM4 GPUs. Note that our choice of batch size is substantially larger than those commonly adopted in other works while training on the ImageNet (e.g., 128 in [31]). We empirically observe that training with a larger batch size generates images of better quality and clearer class alignment. This is consistent with the proposed CTRL-Multi objective as it explicitly encourages alignment of class distributions, therefore benefiting from a larger batch that better captures overall data distributions. We leave a more rigorous study of the effect of batch size for future work.

Due to the heavy computation of such large batch size, we present the intermediate result obtained at the early iteration 35,000 whereas most existing methods run with significantly larger number of iterations. Nevertheless, the intermediate result already verify the efficacy of our framework. In addition, we present the full version of the comparison with existing generative methods in Table A10. We see the IS and FID scores for CTRL-Multi degraded a little after the finetuning. This is expected as learning a more refined separation and alignment of 10 classes is a more challenging task than 2 classes. This is consistently observed from experiments on other datasets too.

Visualizing feature similarity for CTRL-Multi. We visualize the cosine similarity among features $\boldsymbol{Z}$ of different classes learned from the CTRL-Multi objective in Figure A13. In addition, we provide the visualization of alignment between features $\boldsymbol{Z}$ and decoded features features $\hat{\boldsymbol{Z}}$. These results demonstrate that not only the encoder has already learnt to discriminate between classes, but also the learned $\boldsymbol{Z}$ and $\hat{\boldsymbol{Z}}$ are aligned clearly within each class.

Table A10. Comparison on CIFAR-10, STL-10, and ImageNet. ↑ means higher is better. ↓ means lower is better.

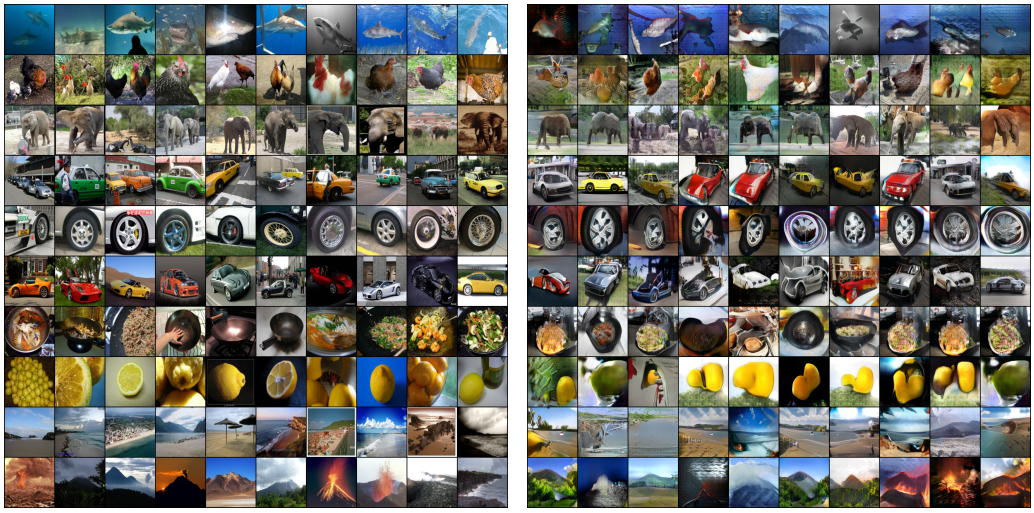
Method	CIFAR-10		STL-10		ImageNet	
	IS↑	FID↓	IS↑	FID↓	IS↑	FID↓
GAN based methods						
DCGAN [63]	6.6	-	7.8	-	-	-
SNGAN [31]	7.4	29.3	**9.1**	40.1	-	48.73
CSGAN [28]	8.1	19.6	-	-	-	-
LOGAN [29]	**8.7**	**17.7**	-	-	-	-
VAE/GAN based methods						
VAE [5]	3.8	115.8	-	-	-	-
VAE/GAN [64]	7.4	39.8	-	-	-	-
NVAE [41]	-	50.8	-	-	-	-
DC-VAE [42]	**8.2**	**17.9**	8.1	41.9	-	-
CTRL-Binary (ours)	**8.1**	**19.6**	8.4	**38.6**	7.74	**46.95**
CTRL-Multi (ours)	7.1	23.9	7.7	45.7	6.44	55.51



(**a**) $|\boldsymbol{Z}^\top \boldsymbol{Z}|$ (**b**) $|\boldsymbol{Z}^\top \hat{\boldsymbol{Z}}|$

Figure A13. Visualizing feature alignment: (**a**) among features $|\boldsymbol{Z}^\top \boldsymbol{Z}|$, (**b**) between features and decoded features $|\boldsymbol{Z}^\top \hat{\boldsymbol{Z}}|$. These results obtained after 200,000 iterations.

Visualizing auto-encoding property for CTRL-Multi. We visualize the original images $\boldsymbol{X}$ and their decoded $\hat{\boldsymbol{X}}$ using the LDR model fine-tuned with the CTRL-Multi objective. The results are shown in Figure A14 for the selected 10 classes in ImageNet. The CTRL-Multi objective can give good visual quality for $\hat{\boldsymbol{X}}$ as well as sample-to-sample alignment.



(**a**) Original $\boldsymbol{X}$ (**b**) Decoded $\hat{\boldsymbol{X}}$

Figure A14. Visualizing the original $\boldsymbol{X}$ and corresponding decoded $\hat{\boldsymbol{X}}$ results on ImageNet (10 classes). The LDR model is fine-tuned using the CTRL-Multi objective. These visualizations are obtained after 35,000 iterations.

Appendix A.8. Ablation Study on Closed-Loop Transcription and Objective Functions

To empirically validate the necessity and respective roles of the closed-loop transcription and the rate reduction (ΔR) objective, we conduct two sets of experiments. For the first set of experiments, we modify our closed-loop architecture by instantiating more than two networks while keeping the objective function (12) unchanged. For the second set of experiments, we keep the closed-loop architecture but replace all rate reduction (ΔR) terms in (12) with corresponding cross-entropy, or remove some of the terms. Experiments here shed insight onto how the closed-loop transcription and the rate reduction affect separately the performance, including sample-wise reconstruction, the alignment of $\boldsymbol{Z}$ and $\hat{\boldsymbol{Z}}$ space, and the diversity of intra-class features.

Appendix A.8.1. The Importance of the Closed-Loop

To evaluate the importance of the closed-loop transcription, we experiment on modified versions of the closed-loop architecture (A3). Notice that many architectures have been proposed and experimented before to promote the encoder f and decoder g to be mutually inverse or cycle consistent (at least for mappings between the data and feature distributions), such as BiGAN [38], VAE-GAN [34], and CycleGAN [56]. However, the cycle consistency is typically enforced through a third discriminator network. (In the case of CycleGAN [56], one needs two additional discriminator networks, one for each domain).

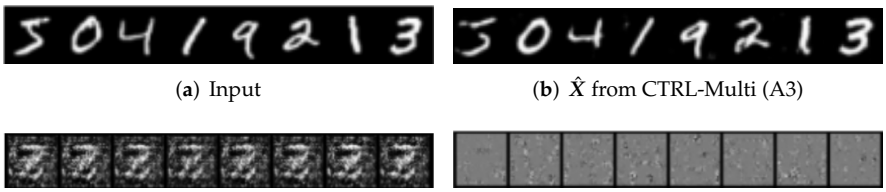
Here, we experiment on whether similar ideas work with the rate-reduction objective. First, we break the closed-loop and use a separate encoder network $f^2 : \hat{\boldsymbol{X}} \to \hat{\boldsymbol{Z}}$ to replace the original encoder f. The revised architecture is summarized in the diagram (A4). Second, to emulate the architecture of VAE-GAN [34], we also instantiate an extra encoder network f^2 and compute the CTRL-Multi objective using $\tilde{\boldsymbol{Z}}$ and $\hat{\boldsymbol{Z}}$. The resulting architecture is also summarized in the diagram (A5).

$$\boldsymbol{X} \xrightarrow{f(\boldsymbol{x},\theta)} \boldsymbol{Z} \xrightarrow{g(\boldsymbol{z},\eta)} \hat{\boldsymbol{X}} \xrightarrow{f(\boldsymbol{x},\theta)} \hat{\boldsymbol{Z}}; \tag{A3}$$

$$\boldsymbol{X} \xrightarrow{f^1(\boldsymbol{x},\theta^1)} \boldsymbol{Z} \xrightarrow{g(\boldsymbol{z},\eta)} \hat{\boldsymbol{X}} \xrightarrow{f^2(\boldsymbol{x},\theta^2)} \hat{\boldsymbol{Z}}; \tag{A4}$$

$$\boldsymbol{X} \xrightarrow{f^1(\boldsymbol{x},\theta^1)} \boldsymbol{Z} \xrightarrow{g(\boldsymbol{z},\eta)} \hat{\boldsymbol{X}}, \boldsymbol{X} \xrightarrow{f^2(\boldsymbol{x},\theta^2)} \hat{\boldsymbol{Z}}, \tilde{\boldsymbol{Z}}. \tag{A5}$$

We run experiments on MNIST with the three different architectures, and choose the network from Table A1 for the encoder and Table A2 for the decoder, and the training hyper-parameters follow Appendix A.1. The qualitative results are shown in Figure A15. Both architectures (A4) and (A5) failed to generate meaningful images. These experiments show that directly applying rate-reduction objectives without the closed-loop or architectures that loosely enforcing cycle consistency fails to work. Instead, the closed-loop formulation allows us to use only two networks, without the need of any extra network.



(**a**) Input (**b**) $\hat{\boldsymbol{X}}$ from CTRL-Multi (A3)

(**c**) $\hat{\boldsymbol{X}}$ from architecture (A4) (**d**) $\hat{\boldsymbol{X}}$ from architecture (A5)

Figure A15. Qualitative results for ablation study with alternative architectures to the proposed CTRL.

Appendix A.8.2. The Importance of Rate Reduction

By replacing the rate reduction (ΔR) terms in the objective function (12) with cross-entropy, we introduce a linear mapping $\boldsymbol{W} \in \mathbb{R}^{d\times k}$ to map $\boldsymbol{Z} \in \mathbb{R}^{d\times n}$ from feature space to logits $\gamma = \boldsymbol{Z}^\top \boldsymbol{W}$. We then calculate the softmax cross-entropy function on logits γ and one hot label matrix $\boldsymbol{Y}$. Here $\mathcal{H}(\gamma, \boldsymbol{Y}) = \sum_{i=1}^{n}\sum_{j=1}^{k} Y_{ij} \log \frac{e^{\gamma_{ij}}}{\sum_{j=1}^{k} e^{\gamma_{ij}}}$ is the formulation of softmax cross-entropy function and $\boldsymbol{Y} \in \mathbb{R}^{n\times k}$ is one hot label matrix. Then, we can replace the first two terms of (12) ($\Delta R(\boldsymbol{Z})$ and $\Delta R(\hat{\boldsymbol{Z}})$) with $\mathcal{H}(\boldsymbol{Z}^\top \boldsymbol{W}, \boldsymbol{Y})$ and $\mathcal{H}(\hat{\boldsymbol{Z}}^\top \boldsymbol{W}, \boldsymbol{Y})$. For the third term of (12), we extract j-th class one hot feature $\gamma_j = \boldsymbol{Z}_j^\top \boldsymbol{W}$, $\hat{\gamma}_j = \hat{\boldsymbol{Z}}_j^\top \boldsymbol{W}$ from $\boldsymbol{Z}$ and $\hat{\boldsymbol{Z}}$, and define the distance $\mathcal{D}(\gamma_j, \hat{\gamma}_j) = \frac{e^{\gamma_j}}{e^{\gamma_j}+e^{\hat{\gamma}_j}}$ of them. For the third term of (12), we further introduce k linear layers as discriminators $\{\mathcal{D}_j\}_{j=1}^{k}$ for each class. Then, we replace the third term with the GAN's objective function as $\sum_{j=1}^{k} \mathbb{E}[\log \mathcal{D}_j(\boldsymbol{Z}_j)] + \mathbb{E}[\log(1 - \mathcal{D}_j(\hat{\boldsymbol{Z}}_j))]$ ($\mathbb{E}[\boldsymbol{X}]$ denote the expectation of $\boldsymbol{X}$). Now, we have the cross-entropy version objective function (A6) for the closed-loop framework. We denote the closed-loop framework with cross-entropy as Closed-loop-CE.

$$\min_{\eta}\max_{\theta,\boldsymbol{W},\mathcal{D}} \mathcal{T}_{\boldsymbol{X}}(\theta, \eta, \boldsymbol{W}, \mathcal{D}) \doteq \mathcal{H}(\boldsymbol{Z}^\top \boldsymbol{W}, \boldsymbol{Y}) + \mathcal{H}(\hat{\boldsymbol{Z}}^\top \boldsymbol{W}, \boldsymbol{Y}) + \sum_{j=1}^{k} \mathbb{E}[\log \mathcal{D}_j(\boldsymbol{Z}_j)] + \mathbb{E}[\log(1 - \mathcal{D}_j(\hat{\boldsymbol{Z}}_j))]. \tag{A6}$$

We run the experiments on MNIST and CIFAR10. The architectures of MNIST and CIFAR10 are given in Tables A1–A4 (In the context of this section, we use the term Decoder and Generator interchangeably; similarly for Encoder and Discriminator).

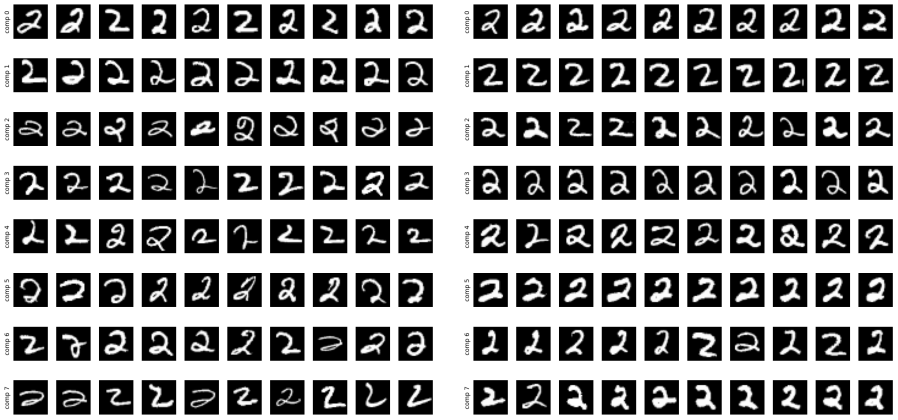
Results on MNIST. The training hyper-parameters of CTRL-Multi and Closed-loop-CE on MNIST are following Appendix A.1. Comparisons between CTRL-Multi and Closed-loop-CE are listed in Figures A16–A18.

Figure A16b,c show the reconstructed images $\hat{\boldsymbol{X}}$ from Closed-loop-CE and CTRL-Multi. Both methods can give sample-wise reconstruction results due to the closed-loop transcription framework. However, comparing training images whose features are best aligned with the principal components of class '2' in Figure A17, we see that the principal components of CE features do not correspond to consistent visual attributes of the images, whereas ours do.

From the heatmaps in Figure A18a,b, we see the features learned by rate reduction possess clear orthogonal subspace structures, whereas those learned by Closed-loop-CE do not. Moreover, Figure A18c,d shows that the learned features of CTRL-Multi have higher singular values for the top principal components of each class, corresponding to a more linearized and diverse feature distribution, whereas those by Closed-loop-CE do not.

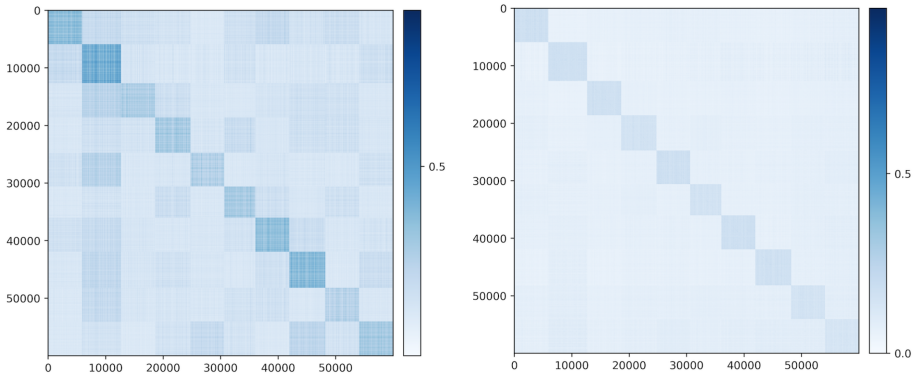
(**a**) Original $\boldsymbol{X}$ (**b**) $\hat{\boldsymbol{X}}$ by Closed-loop-CE (**c**) $\hat{\boldsymbol{X}}$ by CTRL-Multi

Figure A16. The comparison of sample-wise reconstruction between the Closed-loop-CE objective and the CTRL-Multi objective.



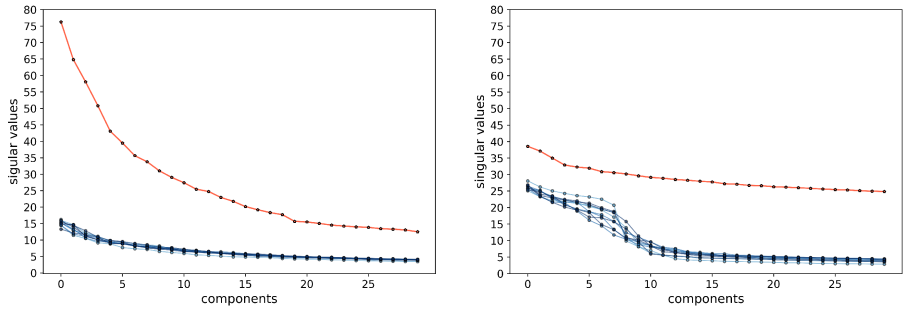
(**a**) Closed-loop-CE (**b**) CTRL-Multi

Figure A17. Training samples along different principal components of the learned features of digit '2'.



(**a**) (**b**)

Figure A18. *Cont.*



(**c**) (**d**)

Figure A18. Comparison Closed-loop-CE and CTRL-Multi on $|\boldsymbol{Z}^\top \hat{\boldsymbol{Z}}|$ and PCA singular values. (**a**) $|\boldsymbol{Z}^\top \hat{\boldsymbol{Z}}|$ from Closed-loop-CE. (**b**) $|\boldsymbol{Z}^\top \hat{\boldsymbol{Z}}|$ from CTRL-Multi. (**c**) PCA of learned features by the Closed-loop-CE objective for each class. (**d**) PCA of learned features by the CTRL-Multi objective for each class.

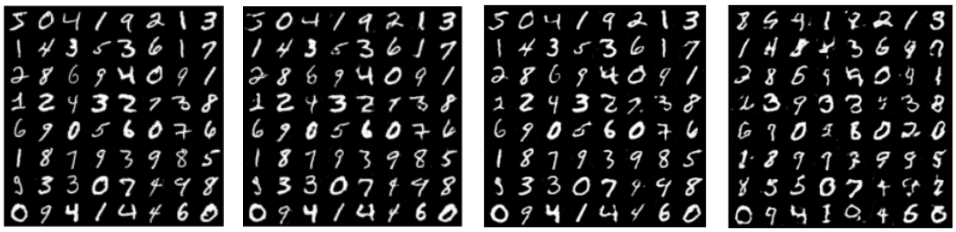
Failed Attempts on CIFAR-10 with Cross Entropy. The training hyper-parameters of Closed-loop-CE on CIFAR10 follow Appendix A.1. We perform the grid search on three hyper-parameters: learning rate $\{1.5 \times 10^{-2}, 1.5 \times 10^{-3}, 1.5 \times 10^{-4}\}$, batch size (800 or 1600), and inner loop (1,2,3,4), conducting 24 experiments in total. All cases of the Closed-loop-CE fail to converge or experience model collapse on the CIFAR-10 dataset.

Appendix A.8.3. Ablation Study on the CTRL-Multi Objectives

In this section, we investigate the influence of each term of the objective function (12) and see how they affect the learned features $\boldsymbol{Z}$, $\hat{\boldsymbol{Z}}$ and sample-wise reconstruction. We follow the same experiment setting with CTRL-Multi on MNIST (Appendix A.1), and conduct three experiments, each with a modified version of the original objective. Objective I is the original objective with all three terms, Objective II removes the second term $\Delta R(\hat{\boldsymbol{Z}})$, and Objective III keeps only the third term $\Delta R(\boldsymbol{Z}, \hat{\boldsymbol{Z}})$. The results in Figure A19 show that using Objective II we can still maintain the sample-wise reconstruction property, but the image quality is lower when compared those constructed by Objective I (Figure A19b vs. Figure A19c). Objective III loses the sample-wise reconstruction property (Figure A19a vs. Figure A19d). Finally, the results from Figures A20 and A21 show that without the first two terms, the learned features $\boldsymbol{Z}$ and $\hat{\boldsymbol{Z}}$ have poor class-to-class alignment and their principal components do not show clear subspace structure with higher singular values within each class.

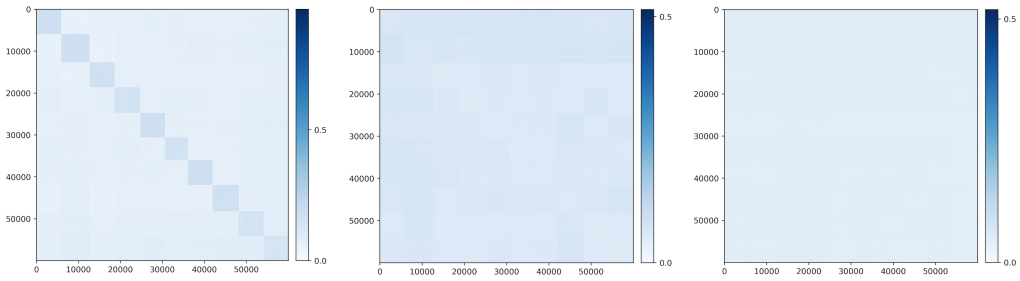
Table A11. Three different objective functions for CTRL.

Objective I:	$\min_\eta \max_\theta \mathcal{T}_{\boldsymbol{X}}(\theta, \eta) = \Delta R\big(\boldsymbol{Z}(\theta)\big) + \Delta R\big(\hat{\boldsymbol{Z}}(\theta, \eta)\big) + \sum_{j=1}^k \Delta R\big(\boldsymbol{Z}_j(\theta), \hat{\boldsymbol{Z}}_j(\theta, \eta)\big)$.
Objective II:	$\min_\eta \max_\theta \mathcal{T}_{\boldsymbol{X}}(\theta, \eta) = \Delta R\big(\boldsymbol{Z}(\theta)\big) + \sum_{j=1}^k \Delta R\big(\boldsymbol{Z}_j(\theta), \hat{\boldsymbol{Z}}_j(\theta, \eta)\big)$.
Objective III:	$\min_\eta \max_\theta \mathcal{T}_{\boldsymbol{X}}(\theta, \eta) = \sum_{j=1}^k \Delta R\big(\boldsymbol{Z}_j(\theta), \hat{\boldsymbol{Z}}_j(\theta, \eta)\big)$.



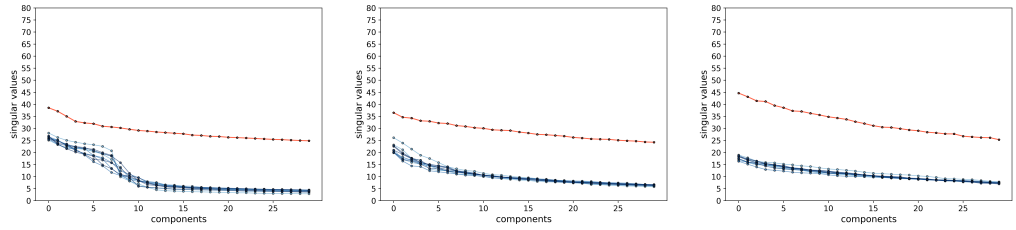
(**b**) Input (**c**) $\hat{X}$ from objective I (**d**) $\hat{X}$ from objective II (**e**) $\hat{X}$ from objective III

Figure A19. The influence of the choice of objective functions on the reconstruction: decoded images $\hat{X}$ from the objective I, II, or III.



(**a**) Objective I (**b**) Objective II (**c**) Objective III

Figure A20. Correlation $|Z^\top \hat{Z}|$ between features Z and $\hat{Z}$ learned with Objective I, II, or III.



(**a**) Objective I (**b**) Objective II (**c**) Objective III

Figure A21. PCAs of the features learned with Objective I, II, or III.

Appendix A.9. Ablation Study on Sensitivity to Spectral Normalization

It is known that spectral normalization is important to improve the stability of training GANs. Here, we test our formulation with and without the spectral normalization. We follow the setting from Appendix A.1 and test on CIFAR10, using the network architecture from Tables A3 andA4. All settings of two experiments are exactly same except with or without spectral normalization. We see that our formulation is stable in both settings and generate similar images. The only difference is that the quantitative scores in terms of IS and FID is higher with the spectral normalization.

Table A12. Ablation study the influence of spectral normalization. ↑ means higher is better. ↓ means lower is better.

Backbone = SNGAN		**CTRL-Binary**		**CTRL-Multi**	
		SN = True	**SN = False**	**SN = True**	**SN = False**
CIFAR-10	IS ↑	8.1	6.6	7.1	5.8
	FID ↓	19.6	27.8	23.9	41.5

Appendix A.10. Ablation Study on Trade-Off between Network Width and Batch Size

Empirically, we observed that for our formulation, the larger the batch size, the better the results. To justify our use of batch size that is larger than those adopted in previous works such as [31], we conduct the following experiment which studies the training behavior of our proposed CTRL-Multi objective. Specifically, we train on the selected 10 classes of ImageNet with varying number of widest channels in our chosen architecture (specified in Appendix A.1) and batch size. We train both the encoder and decoder from scratch without fine-tuning. Other hyper-parameter settings detailed in Appendix A.7 are fixed. We present the results in Table A13. In the table, we denote training sessions that do not produce meaningful images as "failure" and those that do as "success". In the "failure" scenario, we noticed that the second term in the CTRL-Multi objective (12) would collapse to near 0 and could not be recovered, implying the decoder has essentially lost in the minimax game. In the "success" scenario, both the first terms of (12) stay close to each other and neither would collapse to near 0. The results present an interesting diagonal pattern that captures the relationship between batch size and network width. With a wider network and more channels, the network contains a greater capacity but would require a larger batch to stabilize training. This experiment justifies our use of a larger batch in our experiment in Appendix A.7 and also presents an interesting trade-off between network capacity and batch size for training.

Table A13. Ablation study on ImageNet about trade-off between batch size (BS) and network width (Channel #).

	Channel# = 1024	**Channel# = 512**	**Channel# = 256**
BS = 1800	success	success	success
BS = 1600	success	success	success
BS = 1024	failure	success	success
BS = 800	failure	failure	success
BS = 400	failure	failure	failure

Appendix A.11. Ablation Study on Feature Dimension

In this paper so far, for simplicity and uniformity, we have chosen the feature dimension $d = nz$ to be 128 for all experiments. In practice, however, the choice of feature dimension may affect the performance of the learned features: common practices suggest the larger the model, the better the performance could be. Hence, in this last section, we conduct experiments to show how the feature dimension affects the performance. It is not our intention to find the best feature dimension (nor the best network) with this work. We only want to show that there is room to improve the results presented in this paper.

The baseline experiment is conducted on CIFAR-10 with architectures from Table A2 and Table A1, training hyper-parameters are following the setting in Appendix A.1. Here, we change the feature dimension nz, batch size, and learning rate to 512, 8196, and 0.5×10^{-4} respectively. Figure A22 shows the comparison of (randomly selected, not cherry-picked) reconstructed images with the original ones. We observe a significant improvement in visual quality over the results with a lower feature dimension. The IS and FID scores reported in Table A14 also confirm the improvement.

Table A14. IS and FID scores of images reconstructed by LDR models learned with different feature dimensions. ↑ means higher is better. ↓ means lower is better.

		dim = 128		**dim = 512**	
		CTRL-Binary	**CTRL-Multi**	**CTRL-Binary**	**CTRL-Multi**
CIFAR-10	IS ↑	8.1	7.1	8.4	8.2
	FID ↓	19.6	23.6	18.7	20.5

Figure A22. Reconstruction results by LDR models learned with different feature dimensions.

References

1. Lee, J.M. *Introduction to Smooth Manifolds*; Springer: Berlin/Heidelberg, Germany, 2002.
2. Chan, K.H.R.; Yu, Y.; You, C.; Qi, H.; Wright, J.; Ma, Y. ReduNet: A White-box Deep Network from the Principle of Maximizing Rate Reduction. *arXiv* **2021**, arXiv:2105.10446.
3. Kramer, M.A. Nonlinear principal component analysis using autoassociative neural networks. *AICHE J.* **1991**, *37*, 233–243. [CrossRef]
4. Hinton, G.E.; Zemel, R.S. Autoencoders, Minimum Description Length and Helmholtz Free Energy. In Proceedings of the 6th International Conference on Neural Information Processing Systems (NIPS'93), Siem Reap, Cambodia, 13–16 December 1993; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 1993; pp. 3–10.
5. Kingma, D.P.; Welling, M. Auto-encoding variational Bayes. *arXiv* **2013**, arXiv:1312.6114.
6. Zhao, S.; Song, J.; Ermon, S. InfoVAE: Information maximizing variational autoencoders. *arXiv* **2017**, arXiv:1706.02262.
7. Wang, Z.; Bovik, A.; Sheikh, H.; Simoncelli, E. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef] [PubMed]
8. Tu, Z. Learning Generative Models via Discriminative Approaches. In Proceedings of the Conference on Computer Vision and Pattern Recognition, Minneapolis, MN, USA, 18–23 June 2007; pp. 1–8. doi: 10.1109/CVPR.2007.383035. [CrossRef]
9. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2014; pp. 2672–2680.
10. Arjovsky, M.; Chintala, S.; Bottou, L. Wasserstein generative adversarial networks. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 214–223.
11. Salmona, A.; Delon, J.; Desolneux, A. Gromov-Wasserstein Distances between Gaussian Distributions. *arXiv* **2021**, arXiv:2104.07970.
12. Wright, J.; Ma, Y. *High-Dimensional Data Analysis with Low-Dimensional Models: Principles, Computation, and Applications*; Cambridge University Press: Cambridge, UK, 2021.
13. Yu, Y.; Chan, K.H.R.; You, C.; Song, C.; Ma, Y. Learning Diverse and Discriminative Representations via the Principle of Maximal Coding Rate Reduction. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2020.
14. Bengio, Y.; Courville, A.; Vincent, P. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 1798–1828. [CrossRef]
15. Srivastava, A.; Valkoz, L.; Russell, C.; Gutmann, M.U.; Sutton, C. VeeGAN: Reducing mode collapse in GANs using implicit variational learning. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2017; pp. 3310–3320.
16. Mirza, M.; Osindero, S. Conditional generative adversarial nets. *arXiv* **2014**, arXiv:1411.1784.
17. Sohn, K.; Lee, H.; Yan, X. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2015; pp. 3483–3491.
18. Mathieu, M.F.; Zhao, J.J.; Zhao, J.; Ramesh, A.; Sprechmann, P.; LeCun, Y. Disentangling factors of variation in deep representation using adversarial training. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2016; pp. 5040–5048.
19. Van den Oord, A.; Kalchbrenner, N.; Espeholt, L.; Vinyals, O.; Graves, A.; Kavukcuoglu, K. Conditional image generation with PixelCNN decoders. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2016; pp. 4790–4798.
20. Wang, T.C.; Liu, M.Y.; Zhu, J.Y.; Tao, A.; Kautz, J.; Catanzaro, B. High-resolution image synthesis and semantic manipulation with conditional GANs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 8798–8807.
21. Chen, X.; Duan, Y.; Houthooft, R.; Schulman, J.; Sutskever, I.; Abbeel, P. InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2016; pp. 2172–2180.
22. Tang, S.; Zhou, X.; He, X.; Ma, Y. Disentangled Representation Learning for Controllable Image Synthesis: An Information-Theoretic Perspective. In Proceedings of the 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021; pp. 10042–10049. [CrossRef]
23. Li, K.; Malik, J. Implicit Maximum Likelihood Estimation. *arXiv* **2018**, arXiv:1809.09087.

24. Li, K.; Peng, S.; Zhang, T.; Malik, J. Multimodal Image Synthesis with Conditional Implicit Maximum Likelihood Estimation. *Int. J. Comput. Vis.* **2020**, *128*, 2607–2628. [CrossRef]
25. Odena, A.; Olah, C.; Shlens, J. Conditional image synthesis with auxiliary classifier GANs. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 2642–2651.
26. Dumoulin, V.; Shlens, J.; Kudlur, M. A learned representation for artistic style. *arXiv* **2016**, arXiv:1610.07629.
27. Brock, A.; Donahue, J.; Simonyan, K. Large scale GAN training for high fidelity natural image synthesis. *arXiv* **2018**, arXiv:1809.11096.
28. Wu, Y.; Rosca, M.; Lillicrap, T. Deep compressed sensing. In Proceedings of the International Conference on Machine Learning, PMLR, Long Beach, CA, USA, 9–15 June 2019; pp. 6850–6860.
29. Wu, Y.; Donahue, J.; Balduzzi, D.; Simonyan, K.; Lillicrap, T. Logan: Latent optimisation for generative adversarial networks. *arXiv* **2019**, arXiv:1912.00953.
30. Papyan, V.; Han, X.; Donoho, D.L. Prevalence of Neural Collapse during the terminal phase of deep learning training. *arXiv* **2020**, arXiv:2008.08186.
31. Miyato, T.; Kataoka, T.; Koyama, M.; Yoshida, Y. Spectral Normalization for Generative Adversarial Networks. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
32. Lin, Z.; Khetan, A.; Fanti, G.; Oh, S. Pacgan: The power of two samples in generative adversarial networks. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2018; pp. 1498–1507.
33. Feizi, S.; Farnia, F.; Ginart, T.; Tse, D. Understanding GANs in the LQG Setting: Formulation, Generalization and Stability. *IEEE J. Sel. Areas Inf. Theory* **2020**, *1*, 304–311. [CrossRef]
34. Larsen, A.B.L.; Sønderby, S.K.; Larochelle, H.; Winther, O. Autoencoding beyond pixels using a learned similarity metric. *arXiv* **2015**, arXiv:1512.09300.
35. Rosca, M.; Lakshminarayanan, B.; Warde-Farley, D.; Mohamed, S. Variational Approaches for Auto-Encoding Generative Adversarial Networks. *arXiv* **2017**, arXiv:1706.04987.
36. Bao, J.; Chen, D.; Wen, F.; Li, H.; Hua, G. CVAE-GAN: Fine-grained image generation through asymmetric training. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2745–2754.
37. Huang, H.; He, R.; Sun, Z.; Tan, T.; Li, Z. IntroVAE: Introspective Variational Autoencoders for Photographic Image Synthesis. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2018; Volume 31.
38. Donahue, J.; Krähenbühl, P.; Darrell, T. Adversarial feature learning. *arXiv* **2016**, arXiv:1605.09782.
39. Dumoulin, V.; Belghazi, I.; Poole, B.; Mastropietro, O.; Lamb, A.; Arjovsky, M.; Courville, A. Adversarially learned inference. *arXiv* **2016**, arXiv:1606.00704.
40. Ulyanov, D.; Vedaldi, A.; Lempitsky, V. It takes (only) two: Adversarial generator-encoder networks. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
41. Vahdat, A.; Kautz, J. Nvae: A deep hierarchical variational autoencoder. *arXiv* **2020**, arXiv:2007.03898.
42. Parmar, G.; Li, D.; Lee, K.; Tu, Z. Dual contradistinctive generative autoencoder. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 21–24 June 2021; pp. 823–832.
43. Bacharoglou, A. Approximation of probability distributions by convex mixtures of Gaussian measures. *Proc. Am. Math. Soc.* **2010**, *138*, 2619–2619. [CrossRef]
44. Hastie, T. *Principal Curves and Surfaces*; Technical Report; Stanford University: Stanford, CA, USA, 1984.
45. Hastie, T.; Stuetzle, W. Principal Curves. *J. Am. Stat. Assoc.* **1987**, *84*, 502–516. [CrossRef]
46. Vidal, R.; Ma, Y.; Sastry, S. *Generalized Principal Component Analysis*; Springer: Berlin/Heidelberg, Germany, 2016.
47. Ma, Y.; Derksen, H.; Hong, W.; Wright, J. Segmentation of multivariate mixed data via lossy data coding and compression. *PAMI* **2007**, *29*, 9. [CrossRef]
48. Jolliffe, I. *Principal Component Analysis*; Springer: New York, NY, USA, 1986.
49. Hong, D.; Sheng, Y.; Dobriban, E. Selecting the number of components in PCA via random signflips. *arXiv* **2020**, arXiv:2012.02985.
50. Farnia, F.; Ozdaglar, A.E. GANs May Have No Nash Equilibria. *arXiv* **2020**, arXiv:2002.09124.
51. Dai, Y.H.; Zhang, L. Optimality Conditions for Constrained Minimax Optimization. *arXiv* **2020**, arXiv:2004.09730.
52. Korpelevich, G.M. The extragradient method for finding saddle points and other problems. *Matecon* **1976**, *12*, 747–756.
53. Fiez, T.; Ratliff, L.J. Gradient Descent-Ascent Provably Converges to Strict Local Minmax Equilibria with a Finite Timescale Separation. *arXiv* **2020**, arXiv:2009.14820.
54. Bai, S.; Kolter, J.Z.; Koltun, V. Deep Equilibrium Models. *arXiv* **2019**, arXiv:1909.01377.
55. Ghaoui, L.E.; Gu, F.; Travacca, B.; Askari, A. Implicit Deep Learning. *arXiv* **2019**, arXiv:1908.06315.
56. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2223–2232.
57. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
58. Krizhevsky, A.; Hinton, G. Learning Multiple Layers of Features from Tiny Images. 2009. Available online: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf (accessed on 9 February 2022).
59. Coates, A.; Ng, A.; Lee, H. An analysis of single-layer networks in unsupervised feature learning. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, Ft. Lauderdale, FL, USA, 11–13 April 2011; pp. 215–223.

60. Liu, Z.; Luo, P.; Wang, X.; Tang, X. Deep Learning Face Attributes in the Wild. In Proceedings of the International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015.
61. Yu, F.; Seff, A.; Zhang, Y.; Song, S.; Funkhouser, T.; Xiao, J. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv* **2015**, arXiv:1506.03365.
62. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
63. Radford, A.; Metz, L.; Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv* **2015**, arXiv:1511.06434.
64. Larsen, A.B.L.; Sønderby, S.K.; Larochelle, H.; Winther, O. Autoencoding beyond pixels using a learned similarity metric. In Proceedings of the International Conference on Machine Learning, PMLR, New York City, NY, USA, 19–24 June 2016; pp. 1558–1566.
65. Salimans, T.; Goodfellow, I.; Zaremba, W.; Cheung, V.; Radford, A.; Chen, X. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems*, MIT Press: Cambridge, MA, USA, 2016; pp. 2234–2242.
66. Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; Hochreiter, S. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2017; pp. 6626–6637.
67. Jonathan Bennett, J.; Carbery, A.; Christ, M.; Tao, T. The Brascamp-Lieb Inequalities: Finiteness, Structure and Extremals. *Geom. Funct. Anal.* **2007**, *17*, 1343–1415. [CrossRef]
68. Ditria, L.; Meyer, B.J.; Drummond, T. OpenGAN: Open Set Generative Adversarial Networks. *arXiv* **2020**, arXiv:2003.08074.
69. Fiez, T.; Ratliff, L.J. Local Convergence Analysis of Gradient Descent Ascent with Finite Timescale Separation. In Proceedings of the International Conference on Learning Representations, Virtual, 3–7 May 2021.
70. Härkönen, E.; Hertzmann, A.; Lehtinen, J.; Paris, S. Ganspace: Discovering interpretable GAN controls. *arXiv* **2020**, arXiv:2004.02546.
71. Wu, Z.; Baek, C.; You, C.; Ma, Y. Incremental Learning via Rate Reduction. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021.
72. Tong, S.; Dai, X.; Wu, Z.; Li, M.; Yi, B.; Ma, Y. Incremental Learning of Structured Memory via Closed-Loop Transcription. *arXiv* **2022**, arXiv:2202.05411.
73. Lee, K.S.; Town, C. Mimicry: Towards the Reproducibility of GAN Research. *arXiv* **2020**, arXiv:2005.02494.
74. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.