Manifold Learning Benefits GANs

Yao Ni*,†, Piotr Koniusz‡§, Richard Hartley‡,*, Richard Nock♠♣,†
†The Australian National University  ‡Data61/CSIRO  ♠Google Research
firstname.lastname@anu.edu.au

Abstract

In this paper¹, we improve Generative Adversarial Networks by incorporating a manifold learning step into the discriminator. We consider locality-constrained linear and subspace-based manifolds², and locality-constrained non-linear manifolds. In our design, the manifold learning and coding steps are intertwined with layers of the discriminator, with the goal of attracting intermediate feature representations onto manifolds. We adaptively balance the discrepancy between feature representations and their manifold view, which is a trade-off between denoising on the manifold and refining the manifold. We find that locality-constrained non-linear manifolds outperform linear manifolds due to their non-uniform density and smoothness. We also substantially outperform state-of-the-art baselines.

1. Introduction

Generative Adversarial Networks (GANs) [13] are powerful models for image generation [5,19,22], sound generation [11], image stylization [32] and destylization [46–49], super-resolution [62], feature generation [60, 65], etc. The original GAN learns to generate images [5,19,21,22,56] by performing the following min-max game:

\[ \min_{\theta_D} \max_{\theta_G} J(D; \theta_D, G; \theta_G), \tag{1} \]

where \( J() = E_{x \sim p_x(x)} \log(D(x; \theta_D)) + E_{z \sim p_z(z)} \log(1 - D(G(z))) \). Eq. (1) updates parameters \( \theta_D \) of discriminator \( D(x; \theta_D) \) to discriminate between samples from the data distribution \( p_x(x) \) and generative distributions \( p_G(G) \). Simultaneously, parameters \( \theta_G \) of generator \( G(z; \theta_G) \) are updated to fool the discriminator \( D \). Thus, the noise distribution \( p_z(z) \) becomes mapped to \( p_x(x) \) via generator \( G \).

However, GANs typically suffer from three problems: 1) the training instability [25], 2) the so-called mode collapse [44], and 3) overfitting of the discriminator [58].

¹Equal contribution. ²Brain team (richardnock@google.com).

The training instability is an imbalanced competition of the generator and the discriminator due to non-overlapping support between the model distribution and the data distribution [6, 19, 25], leading to poor quality of generated data. Mode collapse has to do with sharply rising gradient around undesirable local equilibria [25, 44], resulting in generation of the same image. Finally, discriminator overfitting leads to excessive memorization and poor generalization.

Indeed, with an excessive number of parameters, the discriminator may memorize the training data instead of learning a meaningful distribution, leading to a high real/fake classification accuracy on the training dataset and a low accuracy on the validation split [5, 20, 66]. Webster et al. [58] argue such a phenomenon mainly affects the discriminator and is undetectable in the generator, with the exception of hybrid adversarial and non-adversarial methods [4] which impose the so-called consistency loss on a generator.

We also observed discriminator overfitting in baseline models e.g., doubling the number of parameters of discriminator resulted in training and validation FID scores of baseline GANs diverging at some intermediate training stage.

Thus, to reduce overfitting of the discriminator, we propose a data-driven feature manifold-learning step and intertwine it with layers of the discriminator. In this way, the discriminator learns the feature manifold at different levels of object abstraction, from fine to coarse, which limits
the complexity of parameter space and separates the signal from noise as both generated and real data are expressed on a common manifold. As a result, the generator diversifies from noise as both generated and real data are expressed on the complexity of parameter space and separates the signal.

Our contributions are threefold:

i. We intertwine locality-constrained and subspace-based feature encoding and dictionary learning steps [29, 34] with blocks of the GAN discriminator to exploit manifold learning in an end-to-end scenario.

ii. We employ a balancing term to help blocks of the discriminator learn the data-driven manifold from the encoder intertwined with them, permitting some degree of freedom in the vicinity of that manifold (§2).

iii. We show that locality-constrained soft assignment coding (the best coder in our experiments) acts as a locally flexible denoiser [1] due to its Lipschitz continuity which we control to vary its operating mode between the ordinary k-means quantization and locality-constrained linear coding. This setting admits quantization of some feature space parts while approximately preserving linearity of other feature space parts (§5).

For contribution (i), we investigate Sparse Coding (SC) [31, 61], Non-negative Sparse Coding (SC+.) [16], Orthogonal Matching Pursuit (OMP) [8, 42], Locality-constrained Linear Coding (LLC) [55], Soft Assignment (SA) [3, 54], and Locality-constrained Soft Assignment (LCSA) [26–29, 34], and Hard Assignment (HA) [7, 51]. We provide formulations and discussion on properties of each coder in §4.

2. Problem Formulation

Figure 1 shows our pipeline (we skip conditional cues for brevity). We build on BigGAN [5], OmniGAN [69], MSG-StyleGAN [18], StyleGAN2 [22] but we equip the discriminator with the manifold learner which is metacontrolled to reduce overfitting. In §C of the supplementary material, we also study the combination of our method with DA [66], ADA [20] and LeCamGAN [53] in limited data scenario.

The discriminator $D(x; \theta_D)$ of the GAN in Eq. (1) classifies input images as real or fake. Many architectures exist in the literature e.g., GAN [13] uses the discriminator based on a convolutional network, whereas recent architectures e.g., BigGAN [5], OmniGAN [69], MSG-StyleGAN [18] and StyleGAN2 [22] use residual discriminators with $L$ residual blocks e.g., see BigGAN [5] (their Fig. 16).

Let $f : \mathbb{R}^{d \times N} \times \mathbb{R}^{d \times N'} \rightarrow \mathbb{R}^{d' \times N'}$ be a function realized by the size of the set of parameters $\theta_B$ be a function realized by a single discriminator block with parameters $\theta_B$, where $d$ and $d'$ are the number of input/output channels, $N = WH$ and $N' = W'H'$ are the number of input/output spatial locations in feature maps of the block. Often, $N'$ may equal $N$.

We introduce an encoding function $h : \mathbb{R}^{d} \rightarrow \mathbb{R}^{d'}$ of small volume in $\mathbb{R}^{d 	imes N'}$, which we sometimes refer to as the feature space. In the cases we consider this mapping derives from a mapping $\mathbb{R}^{d} \rightarrow \mathbb{R}^{d'}$ applied independently and equally over the second dimension $\mathbb{R}^{d' \times N'}$. The encoding introduces an error, measured by $\|h(X) - X\|_F \leq \epsilon$ where $\epsilon$ is called the reconstruction error.

In dictionary-based encoding, the function $h$ relies on a dictionary $M = [m_1, \cdots, m_k] \in \mathbb{R}^{d' \times k}$ containing $k$ column vectors, the so-called dictionary atoms (sometimes called anchors), defining the underlying manifold $\mathcal{M}$, and $k \gg d'$ ensures the dictionary is overcomplete. Then, after solving the optimization problem,

$$(\alpha, M) = \arg \min_{\alpha, M'} \|X - M'\alpha\|_F^2 + \kappa \Omega(\alpha', M', X),$$

where $\alpha \equiv [\alpha_1, \cdots, \alpha_N] \in \mathbb{R}^{d \times N}$, the function $h$ is defined by $h(X) = M\alpha$. Since $\alpha$ depends on $X$, we shall commonly write it as $\alpha(X)$. The mapping $h$ maps $\mathbb{R}^d$ into a subset $M$ of $\mathbb{R}^{d'}$, that we will call the feature manifold (or simply manifold).

The choice of $\Omega(\alpha', M', X)$ realizes some desired constraints via regularization (with $\kappa > 0$) for example, $\Omega(\alpha', M', X) = \|\alpha\|_1$ encourages sparsity of $\alpha$, while $\Omega(\alpha', M', X) = \sum_m \|x_n - m_i\|_2^2, \cdots, \|x_n - m_k\|_2^2\| \alpha_n\|_1$ encourages locality to express each $\alpha_n$ w.r.t. Span$(m_1, \cdots, m_k)$, where $m_1, \cdots, m_k$ are the $k'$ nearest neighbors of $x_n$.

We intertwine the encoding step with blocks of the discriminator as follows:

$$X_{i+1} = (1 - \beta)\tilde{X}_i + \beta h_i(\tilde{X}_i)$$

where $\tilde{X}_i = f(X_i; \theta_{B_i})$ and $h_i$ is the encoding function introduced just above, expressed in terms of a dictionary $M_i$, whereas $\{\theta_{B_i}\}_{i=1}^L$ and $\{M_i\}_{i=1}^L$ are parameters of blocks of the discriminator and dictionaries for layers $1, \ldots, L$ respectively. Figure 2 illustrates Eq. (3) applied to blocks $B_1, \cdots, B_L$ of the discriminator.
where

\[ \beta \]

and refine dictionary \( \tilde{M} \) from \( \tilde{M} \) and then recovered via

\[ h \]

and their views recovered from manifolds, and the smooth-

\[ h \]

collapse. Disconnected Manifold GAN [23] assumes that

\[ h \]

match the generated and real data distributions with first- and second-order statistics. A Maximum Mean Discrepancy GAN [33] matches distributions in the Reproducing Kernel Hilbert Space (RKHS).

Spectral Normalization GAN [36] applies normalization on weights to stabilize the discriminator. Spectral Regularization GAN [35] performs detection of so-called spectral collapse. Disconnected Manifold GAN [23] assumes that natural images lie on a union of disjoint manifolds. Feature Quantization GAN (FQGAN) [67] quantizes features of discriminator into a k-means based dictionary. The Denoising Feature Matching GAN [57] encourages proximity between the output of the generator and a denoising auto-encoder.

Our work differs from the Disconnected Manifold GAN which models an entire image distribution as a union of non-explicit manifolds (a collection of generators). FQGAN imposes quantization on features of the discriminator and Denoising Feature Matching GAN learns a denoising auto-encoder on real images to apply it on generated images.

In contrast, we model coarse-to-fine features extracted from multiple blocks of the discriminator, which capture different semantic levels of abstraction. We encourage these features to lie on explicit locality-constrained non-linear manifolds (each block of our discriminator has its own learner). We adaptively control the mixing levels of features and their views recovered from manifolds, and the smoothness of manifolds to prevent overfitting.

**Limiting Overfitting.** Augmentations (rotations, clipping) [17, 52, 64, 66] can limit overfitting, however, augmentation
4. Preliminaries

Below, we explain GAN pipelines on which we build, and feature encoding and dictionary learning, our key tools.

4.1. Baseline GANs

BigGAN [5] combines a projection-based loss [37], spectral normalization [36], and self-attention [63]. The projection-based score is a trade-off between a class-wise cosine similarity and a class independent term:

\[ s(x, y) = y^T V D(x; \theta_D) + f'(D(x; \theta_D); \theta_{D'}) \]  

(7)

where \( y \in \{0, 1\}^C \) and \( \|y\|_1 = 1 \). \( V \in \mathbb{R}^{C \times d} \) is a bilinear compatibility matrix that associates output features \( X_{\text{out}} \) from \( D \) with the class label. As \( y \) is a one-hot vector, \( V \equiv \{v_1, \ldots, v_C\}^T \) contains linear projectors \( v_c \), one per class \( c \in \{0, \ldots, C-1\} \). Function \( f' : \mathbb{R}^d \rightarrow 1 \) is realized by an FC layer with parameters \( \theta_{D'} \). Scores \( s(x, y) \) are passed to a hinge-based loss with two components:

\[ J_{\text{disc}} = E_{(x, y) \sim p_{\mathcal{D} \mathcal{W}}(x, y)} \max(0, 1 - s(x, y)) + 
E_{(z_1, y) \sim p_{\mathcal{D} \mathcal{W}}(z, y)} s(G(z, y), y) \].

(8)

Our manifold-based pipeline combines loss \( J_{\text{prox}} \) from Eq. (4) with \( J_{\text{disc}} \) and the original \( J_{\text{gen}} \) from [5].

Omnigan [69] uses a multi-label softmax loss (where a label vector \( y \in \{0, 1\}^{C+2} \) is a concatenation of the one-hot class label vector and one-hot real/fake vector):

\[ J_{\text{disc}} = E_{(x, y) \sim p_{\mathcal{D} \mathcal{W}}(x, y)} s(x, y) + 
E_{(z_1, y) \sim p_{\mathcal{D} \mathcal{W}}(z, y)} s(G(z, y), y) \]

\[ s(x, y) = \sum_{c=0}^{C+1} \log \left( 1 + e^{-\text{Sign}(y_c - 0.5) \phi_\alpha(x)} \right). \]

(9)

We note that \( X_{\text{out}} \) represents output features from \( D \), \( \phi_\alpha(x) = f'(D(x; \theta_D); \theta_{D'}) \), where function \( f' : \mathbb{R}^d \rightarrow C+2 \) is realized by an FC layer with parameters \( \theta_{D'} \).

4.2. Feature Coding and Dictionary Learning on Data-driven Manifolds

Below, we formalize feature encoding and dictionary learning approaches listed in §1. In experiments, we substitute a chosen coding step into function \( h(x) \) from §2. Moreover, \( M \equiv \{m_1, \ldots, m_k\} \in \mathbb{R}^d \times k \) is a dictionary whose learning step is detailed at the bottom of §4.2, and \( \alpha(x) \) represents the encoding/mapping on the simplex.

Hard Assignment (HA) [7, 51]. This encoder assigns each \( x \) to its nearest \( m \) by solving the following optimisation problem:

\[ \alpha(x) = \arg \min_{\alpha \in \{0, 1\}^k} \|x - M\alpha\|_2^2, \]

\[ \text{s.t. } \|\alpha\|_1 = 1. \]

(11)

If \( M \) is formed by k-means clustering, HA becomes an equivalent of the quantizer from FQGAN [67].

Sparse Coding (SC) [31, 61] & Non-negative Sparse Coding (SC+) [16]. SC encodes \( x \) as a sparse linear combination of atoms \( M \) by optimising the following objective:

\[ \alpha(x) = \arg \min_{\alpha'} \|x - M\alpha\|_2^2 + \kappa \|\alpha\|_1, \]

(12)

whereas SC+ additionally imposes a constraint that \( \alpha' \geq 0 \). Both SC and SC+ encode \( x \) on a subset of \( M \) of size controlled by the sparsity term.

Orthogonal Matching Pursuit (OMP) [8, 42]. This encoder expresses \( x \) as a sparse linear combination of atoms \( M \) by optimising the following objective:

\[ \alpha(x) = \arg \min_{\alpha'} \|x - M\alpha\|_2^2, \]

\[ \text{s.t. } \|\alpha'\|_0 \leq \tau, \]

(13)

where the pseudo-norm \( \|\alpha\|_0 \) ensures the count of non-zero coefficients of \( \alpha' \) is at most \( \tau \). Unlike SC and SC+, \( \|\alpha'\|_0 \), the penalty enforces a strict limit on the number of non-zero elements in \( \alpha' \), but the problem itself is NP-hard.

Approximate Locality-constrained Linear Coding (LLC) [55]. LLC expresses \( x \) as a linear combination of \( k' \) nearest neighbor atoms of \( x \) selected from \( M \), forming subspaces of size \( k' \) on a piecewise-linear manifold:

\[ \alpha(x) = \arg \min_{\alpha'} \|x - M\alpha\|_2^2, \]

\[ \text{s.t. } 1^T \alpha' = 1, \]

(14)

and \( \alpha' \) is further constrained by \( \alpha'_i = 0 \) unless \( m_i \) is one of the \( k' \) closest neighbors of \( x \).

Soft Assignment (SA) [3, 54] & Locality-constrained Soft Assignment (LCSA) [26, 29, 34]. SA expresses \( x \) as the membership probability (concept known from GMM [3]) of \( x \) belonging to each \( m_i \) in \( M \) under equal mixing probability and equal variance \( \sigma \) of GMM. SA is given as:

\[ \alpha(x; M, \sigma) = S_\sigma(\|x - m_1\|_2^2, \ldots, |m_k - m_k|_2^2), \]

(15)

where \( S_\sigma \) is the softmax function \( S_\sigma : \mathbb{R}^k \rightarrow \Delta^{k-1} \), where \( \Delta^{k-1} \) is the probability simplex and:

\[ S_\sigma(d_1, \ldots, d_k)_j = \frac{\exp(-d_j^2/2\sigma^2)}{\sum_i \exp(-d_i^2/2\sigma^2)}. \]

(16)

This model yields largest values of \( \alpha'_j \) for atoms \( m_i \) in \( M \) that are close Euclidean neighbors of \( x \). However, \( \alpha_i(x) > 0 \) even for \( m_i \) that are far from \( x \). For this reason, SA is not strictly locality-constrained.

LCSA differs from SA by setting \( \alpha_i(x) = 0 \) unless \( m_i \) is among the \( k' \) nearest-neighbor atoms for \( x \). The denominator of Eq. (16) performs normalization, that is the summation runs over the \( k' \) nearest neighbors. Thus, LCSA maps \( \mathbb{R}^d \) onto a set of probability simplices \( \Delta^{k-1} \). As LCSA was the best in our experiments, we analyze it in §5.

Dictionary Learning (DL). For the above coders, we employ a class-agnostic dictionary learning objective which
follows Eq. (2). Let some \( \alpha(X) = [\alpha_1, \ldots, \alpha_N] \), then:
\[
M = \arg \min_{M'} \| X - M' \alpha \|_F^2, \tag{17}
\]
where \( M' \) can be constrained to contain atoms \( \| m' \|_2 \leq 1 \) if codes \( \alpha \) have non-restricted \( L_2 \) norm e.g., for OMP.

**Inverse of \( \alpha \).** To reproject \( \alpha(X) \) from the manifold \( M \) into the Euclidean space, we simply compute \( \hat{X} = M \alpha(X) \).

**Implementation Remarks.** Coding methods, dictionary learning, and their implementations are detailed in §J of the supplementary material. For dictionary learning, we detach \( X \) and \( \alpha \), and run 1 iteration of gradient descent per mini-batch w.r.t. each \( M \) (no big gain for \( \geq 2 \) iterations). For SC and SC+, we detach \( X \) and all \( M \), and let 5 iterations of gradient descent (no gain for \( \geq 6 \) iterations). LLC has a closed-form solver [55]. Our efficient OMP solves the system of linear equations (no matrix inversion). SA/LCSA enjoy a fast closed-form recipe. LLC and LCSA use the partial sort algorithm for selecting \( k' \) nearest neighbors. We detach \( M \alpha \) to compute the proximity loss in Eq. (4).

5. Theoretical Analysis of LCSA

As LCSA is the best encoder in our experiments, we focus on its theoretical properties below. All proofs for theories listed below are in §L of the supplementary material.

In this section, we use the following notation. Suppose a dictionary of atoms \( M = [m_i] \in \mathbb{R}^d \) is given, and \( 2^k \) denotes the set of all subsets of size \( k \) of \( M \). We define by \( \text{NN}_{k'} : \mathbb{R}^d \to 2^k \) to be the set-valued function that takes a point \( x \) to its set of \( k' \) nearest neighbor atoms. A maximal subset of \( \mathbb{R}^d \) on which \( \text{NN}_{k'}(x) \) is constant is called a Voronoi cell. For a given set \( U \), \( 2^k \), then, a Voronoi cell \( (\text{NN}_{k'})^{-1}(U) \) is a subset of \( \mathbb{R}^d \) consisting of the all the points for which \( U \) is the set of \( k' \) nearest neighbors. The collection of all Voronoi cells constitutes a decomposition of \( \mathbb{R}^d \) into disjoint polyhedral regions.

In the case where the set of \( k' \) nearest elements of \( M \) is not unique, we leave the set \( \text{NN}_{k'}(x) \) undefined. Thus, the Voronoi cells are disjoint open polyhedral regions such that \( \text{NN}_{k'}(x) \) is constant on each cell. The complement in \( \mathbb{R}^d \) of the set of Voronoi cells is a subset of a finite set of hyperplanes in \( \mathbb{R}^d \).

**SA and LCSA encoding.** Given a dictionary \( M = [m_i] \) in \( \mathbb{R}^d \) consisting of \( k' \) elements, with \( k' \leq d + 1 \), we consider the function \( M \alpha(x) \) where \( \alpha(x) = S_{k'}(x) \) is the softmax mapping Eq. (15). (The dictionary may be the dictionary of \( k' \) nearest neighbors of some point \( x \).)

**Proposition 1** If \( \sigma > 0 \), the mapping \( x \mapsto M \alpha(x) \) is a smooth fibration from \( \mathbb{R}^d \) onto the interior of the simplex \( \Delta \) with vertices \( m_i \). The fibre of this mapping is equal to the linear subspace of \( \mathbb{R}^d \) normal to the affine space spanned by the \( m_i \).

A fibre is the set of points that map to the same point in \( \Delta^{k' - 1} \) under this mapping.

**Proposition 2** The following properties of LCSA hold true:

1. If \( \sigma \to 0 \) or \( k' = 1 \), the \( \alpha \) codes converge to the HA solution (quantization).
2. \( \alpha(x) \) is an approximately linear coding of \( x \) in the proximity of \( \mu(x) = \frac{1}{k'} \sum m_i \in \text{NN}_{k'}(x) m_i' \).
3. For \( x \) with nearest neighbor atom \( \{ m_i \} = \text{NN}_{k'}(x) \) with mean \( \mu(x) \), and nearest neighbor atom \( n(x) = \text{NN}_1(x) \), the reconstruction error satisfies \( \| M \alpha(x) - x \|_2 \leq \max \left( \| x - n(x) \|_2, \| x - \mu(x) \|_2 \right) \).
4. The reconstruction error varies smoothly on each Voronoi cell. For \( x \) and \( x' \) in the same Voronoi cell and \( \Delta \) the simplex with vertices \( \text{NN}_{k'}(x) \), we have
   - Local Lipschitz continuity: if \( \sigma > 0 \), then
     \[
     \| M \alpha(x) - M \alpha(x') \| \leq K \| x - x' \|
     \]
     where \( K = D^2 / \sigma^2 \) and \( D \) is diameter of simplex \( \Delta \) (the maximum distance between its vertices). The Lipschitz condition holds for \( \| \cdot \|_1 \) and \( \| \cdot \|_2 \) norms.
   - The biggest change of the reconstruction error on a Voronoi cell for HA (\( \sigma = 0 \)) is less than or equal to \( D \).

5. The LCSA encoding \( M \alpha(x) \) is non-continuous at the boundaries of the Voronoi regions.

**Proposition 3** Our design fulfills the principles of GAN with Denoising Auto-Encoder (DAE) [1] with loss:
\[
\mathcal{L}_{\text{dae}} = \frac{1}{N} \sum_{n=1}^{N} \left( \| r(x_n) - x_n \|_2^2 + \sigma^2 \left\| \frac{\partial r(x)}{\partial x} \right\|_{2} \right)^2, \tag{18}
\]
where \( r(x_n) \) is the reconstruction of \( x_n \), akin to our \( h(x_n) \) in Eq. (4). More importantly, \( \sigma^2 \) specifies the noise variance. Specifically, we relate the following:

6. Not surprisingly, the proximity loss in Eq. (4) fulfills similar role to \( \| r(x_n) - x_n \|_2^2 \) in Eq. (18).

7. LCSA implicitly fulfills the denoising role (apart from locality-constrained non-linear coding) of \( \sigma^2 \| \frac{\partial r(x)}{\partial x} \|_F^2 \), with \( \sigma \) of LCSA and \( \sigma' \) of DAE related i.e., \( \sigma' \) is proportional to our \( \sigma^2 \). To this end, we notice that DAE penalises the Frobenius norm of the Jacobian matrix \( \| \frac{\partial r(x)}{\partial x} \|_F^2 \) by increasing \( \sigma^2 \). We penalise the spectral norm of Jacobian matrix \( \| \frac{\partial M \alpha(x)}{\partial x} \|_2 \) via de facto controlling the Lipschitz constant \( K = D^2 / \sigma^2 \).

**Discussion.** The blue box below explains how properties of LCSA contribute to discriminator training, and how they let LCSA inherit the best properties of other coding methods.
6.1. Network Architecture and Hyper-parameters

We build on OmniGAN/BigGAN/StyleGAN2 for CIFAR-10. For CIFAR-100/ImageNet (64×64), we experiment with OmniGAN/BigGAN. For ImageNet (128×128), we build upon OmniGAN given OmniGAN consistently outperforms BigGAN. We employ MSG-StyleGAN as our baseline for Oxford-102 Flowers. For FFHQ, we build upon StyleGAN2 (see §1 of the supplementary material).

6.2. Results of Image Generation

The generated images for each dataset are given in §K of the supplementary material.

CIFAR-10. Table 1 shows results on OmniGAN+LCSA, which outperforms the baseline OmniGAN by 0.25 and 1.43 on the IS and tFID metrics (d′=256). With d′=512, we outperform OmniGAN by 0.32 and 3.52. We obtain further improvements with d′=1024 while baselines struggle to converge. Comparisons of different models with d′=1024 are in the §B of the supplementary material.

CIFAR-100. Table 2 shows results on OmniGAN+LCSA against the state of the art. For d′=256, our method gains 0.09 and 1.9 on the IS and tFID metrics over the baseline OmniGAN. For d′=512, we outperform OmniGAN by 0.93 and 3.91 (IS and tFID). Further gains are achieved for d′=1024 while other methods struggle to converge. We include TAC-GAN [12] for comparison.

ImageNet (64×64). Table 3 shows that BiGAN+LCSA outperforms BigGAN by 2.97 and 3.01 (tFID & vFID). OmniGAN+LCSA improves OmniGAN by 6.86 and 2.72 (IS & tFID), which is the state of the art in GANs.

ImageNet (128×128). Table 4 shows OmniGAN+LCSA improves OmniGAN by 23.32 and 2.28 (IS & tFID).

Oxford-102 Flowers. Figure 4(a) shows that MSG-StyleGAN+LCSA improves MSG-StyleGAN by 5.46 on FID. Moreover, Figure 4(b) shows that at around iteration...
Table 3. Results on ImageNet (64 x 64). We combine OmniGAN and BigGAN with LCSA. We set \( d' = 384 \).

| Model                | IS ↑ | tFID ↓ | vFID ↓ |
|----------------------|------|--------|--------|
| Inst. Sel. GAN [10]  | 43.30| 9.07   | –      |
| BigGAN               | 34.50| 8.96   | 8.80   |
| FQGAN                | 33.14| 8.27   | 8.15   |
| BigGAN+N+LCSA        | 33.29| 5.99   | 5.79   |
| OmniGAN              | 70.59| 7.09   | 7.66   |
| OmniGAN+LCSA         | 77.45| 4.26   | 4.94   |

Table 4. Results on ImageNet (128 x 128) with \( d' = 384 \). \(^1\) stands for quoting from [69]. \(^\dagger\) are results reproduced by us.

| Model                | IS ↑ | tFID ↓ | vFID ↓ |
|----------------------|------|--------|--------|
| BigGAN\(^\dagger\)   | 104.57| 9.19   | 9.18   |
| OmniGAN\(^\dagger\)  | 190.94| 8.30   | 8.93   |
| OmniGAN\(^\dagger\)  | 169.13| 7.11   | 7.30   |
| OmniGAN+N+LCSA       | 192.45| 4.83   | 5.24   |

Figure 4. Results on Oxford-102 Flowers. (a) FID of different models \(^1\) are results collected from [18]. (b) FID w.r.t. the iteration number for MSG-StyleGAN and MSG-StyleGAN+N+LCSA on Oxford-102 Flowers. Black dots indicate the minimum FID.

| Data      | StyleGAN | +ADA | +LeCam | +bCR | +LCSA | +LCSA+bCR |
|-----------|----------|------|--------|------|-------|----------|
| 70k       | 5.28     | 4.30 | -      | 3.79 | 3.83  | 3.47     |
| 140k      | 3.71     | 3.81 | 3.66   | 3.53 | 3.32  | 3.20     |

Table 5. The FID ↓ results on FFHQ (256 x 256) dataset.

number 75\(K\), MSG-StyleGAN starts diverging while MSG-StyleGAN+LCSA (Ours) continues to decrease FID.

FFHQ (256 x 256). Table 5 shows that StyleGAN2 with LCSA improves the FID of StyleGAN2 by 1.55/0.39 on 70\(K\)/140\(K\) dataset. Moreover, our LCSA can also be combined with bCR [68] to improve the performance.

Data-limited Generation on CIFAR-10/100. We provide comprehensive experiments (10% and 20% data) in §C of the supplementary material. We summarize our findings as: (1) the augmentation-based ADA [20] and DA [66] leak augmentation artifacts to the generator, while ADA+LCSA and DA+LCSA alleviate this issue (see Figure 24 in the supplementary material). (2) LCSA harmonizes with ADA, DA and LeCam loss [53]. (3) We achieve the state of the art on this limited data setting.

6.3. Impact of Hyperparameters

Below, we conduct ablations on CIFAR-10 for \( d' = 512 \).

Nearest neighbors \( k' \), \( \sigma \) and \( k \). Figure 5(a) shows results on OmniGAN+LCSA w.r.t. \( k' \) on CIFAR-100, which verifies that the locality-constrained manifold can be constructed with \( 8 \leq k' \leq 32 \ll k \) with a sufficiently small reconstruction error below the worst case (Prop. 2.3). For ImageNet, \( k' = 8 \) also led to best results. Figure 5(b) verifies that \( 1 \leq \sigma \leq 1.5 \) provides the best trade-off in terms of smoothness (quantization vs. linearization) (Prop. 2.1, 2.2 and 2.4). Figure 5(c) verifies the benefit of dictionary overcompleteness \( d'' \ll k \) as \( k = 1024 \).

Blocks \( l \in \{1, \cdots, L \} \) and LCSA. Figure 5(d) shows the results for various combinations of injection of LCSA into blocks of discriminator. It appears that injecting LCSA into all blocks yields the lowest tFID/vFID, which verifies that constructing multiple manifolds at coarse-to-fine semantic level controls the complexity of the discriminator.

Metaparameter \( \beta \) and impact of \( \eta \). \( \beta \) in Eq. (5) controls the mixing of conv. features obtained by \( f \) and their view \( \eta \) recovered from the manifold, as in Eq. (3). Figure 6(a) shows that in early iterations, blocks of discriminator learn conv. features. Around 25\(K\) iterations, the model starts oscillating between prevention of overfitting and refining conv. features, as indicated by the green curve that continues to gradually grow for \( \eta = 0.5 \) (c.f. flat curve for \( \eta = 0.3 \)). Figure 6(b) also shows that \( \eta = 0.5 \) is a universally good threshold.

6.4. Analysis and Ablation studies

We analyse our method on CIFAR-10/100 \((d' = 512)\). Preventing Discriminator Overfitting. Figure 7 verifies that the discriminator of standard methods yields high real/fake accuracy on training images but low accuracy on testing images. Thus, they overfit to the training set (see the large discrepancy between real and fake images). Thus, standard methods diverge early (see FID) but our method continues learning, as shown in Figure 8 and Figure 4(b).

Impact of Different Components. In Table 6, we ablate (i) dictionary learning (Eq. (17)), (ii) the adaptive mixing input Eq. (3) and (iii) proximity loss Eq. (4). We conduct experiments with settings: (1) ACM: removing our main-fold learner and the proximity loss, and changing Eq. (3) to be \( \mathbf{X}_{t+1} = (1 - \beta)\mathbf{X}_t \) to adaptively control the magnitude of \( \mathbf{X}_t \); (2) LCSA(\( \gamma = 0 \)): removing the proximity loss but keeping the adaptive mixing input; (3) LCSA(\( \beta = 0 \)): removing the adaptive mixing input but keeping the adaptive proximity loss; (4) LCSA(EMA): replacing our dictionary learning with the exponential moving average; (5) LCSA(fixed (\( \beta, \gamma \))): fixed meta-controller (\( \beta, \gamma \)) (different combination of (\( \beta \) and \( \gamma \)) is given in §G of the supplementary material). Table 6 shows that (i) LCSA(\( \gamma = 0 \)) is better than ACM and OmniGAN which verifies the benefit of adaptive mixing input. (ii) LCSA(\( \beta = 0 \)) is better than OmniGAN which verifies the importance of controlling the complexity of learning. (iii) EMA performs worse than our dictionary learning. (iv) As fixed (\( \beta, \gamma \)) scores lower thus the adaptive meta-controller is useful. In §H (supplementary material), LCSA yields larger errors at foregrounds and smaller at backgrounds, thus intertwining step (mixing in-
Figure 5. Ablation studies on CIFAR-100 w.r.t. $k'$ (nearest neighbors), $\sigma$ (controls the Lipschitz constant of LCSA), $k$ (dictionary size), and blocks $l$ which use LCSA. We indicate metrics such as the IS $\uparrow$, tFID $\downarrow$ and vFID $\downarrow$.

Figure 6. Evolution of metaparameter $\beta$ and the impact of $\eta$ which controls the behavior of the detector of overfitting.

Figure 7. Discriminator predictions on CIFAR-10. We plot the output of discriminator on training images and generated fake images. We test Acc (real/fake accuracy predicted by the discriminator) on the training images and testing images. We use different colors to represent different models. Solid/dash/dot lines indicate training/fake/testing images.

Figure 8. Evolution of tFID and vFID for different models on CIFAR-10. Black dots indicate the minimum FID.

put) helps refine/denoise features before input to next layer. **Different Encoders.** Table 6 compares coding methods, each applied to all $l = 1, \ldots, L$ blocks of discriminator. LCSA is the best performer followed by SA and LLC or SC, which validates that using locality-constrained soft as-

Table 6. Results for ablation studies on CIFAR-10 & CIFAR-100. The enhancement whose continuity is controlled via the Lipschitz constant (Prop. 2.4) inverse-proportional to $\sigma^2$ is more robust than locally-linear coding, due to the quantization vs. linear reconstruction trade-off and the role of $\sigma^2$ in denoising (Prop. 3). We also tried replacing the LCSA coder with Denoising Auto-Encoder (DAE) [1] (the setup is in §F of the supplementary material). LCSA achieves better tFID and vFID than locality-constrained linear coding, subspace learning, and DAE. Figure 25 of the supplementary material shows the variance computed over LCSA codes for individual images. While visually mundane regions have low variance (i.e., single code of $\alpha$ encodes it), the visually diverse regions have variance that is somewhat higher for LCSA than other coders, preventing overfitting where it matters.

7. Conclusions

We have applied data-manifold learning (LCSA) in the coarse-to-fine manner to conv. features of discriminator to prevent its overfitting by adaptively balancing the trade-off between denoising on the manifold and refining the manifold (intertwining and dictionary learning steps), controlling the complexity of learning (the proximity loss). Locality-constrained soft assignment is controlled via the Lipschitz constant, resulting in a trade-off between quantization and linear coding, yielding state-of-the-art results.

Acknowledgements. We thank CSIRO’s Machine Learning and Artificial Intelligence Future Science Platform (MLAI FSP) and China Scholarship Council (CSC) for the support.
References

[1] Guillaume Alain and Yoshua Bengio. What regularized auto-encoders learn from the data-generating distribution. *JMLR*, 15(110):3743–3773, 2014. 2, 5, 8, 11, 13

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML*, pages 214–223, 2017. 3

[3] Jeff A Bilmes. A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models. Technical report, 1998. 2, 4

[4] Piotr Bojanowski, Armand Joulin, David Lopez-Pas, and Arthur Szlam. Optimizing the latent space of generative networks. In *ICML*, volume 80, pages 600–609, 2018. 1

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019. 1, 2, 4, 6, 14

[6] Casey Chu, Kentaro Minami, and Kenji Fukumizu. Smoothness and stability in gans. In *ICLR*, 2020. 1

[7] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision*, ECCV, 2004. 2, 4

[8] Geoffrey Davis, Stéphane Mallat, and Zhifeng Zhang. Adaptive time-frequency decompositions with matching pursuits. *Optical Engineering*, 33, 1994. 2, 4

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009. 6, 11

[10] Terrance DeVries, Michal Drozdzal, and Graham W Taylor. Instance selection for gans. *NeurIPS*, 2020. 7

[11] Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial audio synthesis. In *ICLR*, 2019. 1

[12] Mingming Gong, Yanwu Xu, Chunyuan Li, Kun Zhang, and Kayhan Batmanghelich. Twin auxiliary classifiers gan. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *NeurIPS*, pages 1330–1339, 2019. 6

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 1, 2, 3

[14] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. In *NeurIPS*, pages 5769–5779, 2017. 3

[15] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two-time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, volume 30, pages 6626–6637, 2017. 6

[16] Patrik O Hoyer. Non-negative sparse coding. In *Proceedings of the 12th IEEE workshop on neural networks for signal processing*, pages 557–565. IEEE, 2002. 2, 4

[17] Jongheon Jeong and Jinwoo Shin. Training {gan}s with stronger augmentations via contrastive discriminator. In *ICLR*, 2021. 3

[18] Animesh Karanwar and Oliver Wang. Msg-gan: Multi-scale gradients for generative adversarial networks. In *CVPR*, pages 7799–7808, 2020. 2, 3, 6, 7, 14

[19] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018. 1, 3

[20] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *NeurIPS*, 33, 2020. 1, 2, 3, 4, 6, 7, 11, 12, 14

[21] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 1, 3, 6, 22

[22] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, pages 8110–8119, 2020. 1, 2, 6

[23] Mahyar Khayatkhoei, Ahmed Elgammal, and Maneesh Singh. Disconnected manifold learning for generative adversarial networks. In *NeurIPS*, pages 7354–7364, 2018. 3

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014. 14

[25] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. On convergence and stability of gans. *arXiv:1705.07215*, 2017. 1

[26] Piotr Koniusz and Krystian Mikolajczyk. Soft Assignment of Visual Words as Linear Coordinate Coding and Optimisation of its Reconstruction Error. *ICIP*, 2011. 2, 4

[27] Piotr Koniusz, Fei Yan, Philippe-Henri Gosselin, and Krystian Mikolajczyk. Higher-order Occurrence Pooling on Mid-level and Low-level Features: Visual Concept Detection. Technical report, INRIA, Sept. 2013. 2

[28] Piotr Koniusz, Fei Yan, Philippe-Henri Gosselin, and Krystian Mikolajczyk. Higher-order occurrence pooling for bags-of-words: Visual concept detection. *TPAMI*, 2016. 2

[29] Piotr Koniusz, Fei Yan, and Krystian Mikolajczyk. Comparison of Mid-Level Feature Coding Approaches And Pooling Strategies in Visual Concept Detection. *CVIU*, 2012. 2, 4

[30] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009. 6, 11

[31] Honglak Lee, Alex Battle, Rajat Raina, and Andrew Ng. Efficient sparse coding algorithms. *NeurIPS*, pages 801–808, 2006. 2, 4

[32] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *ECCV*, 2016. 1

[33] Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabas Poczos. Mmd gan: Towards deeper understanding of moment matching network. In *NeurIPS*, volume 30, pages 2203–2213, 2017. 3

[34] Liu Lingqiao, Lei Wang, and Xinwang Liu. In Defence of Soft-assignment Coding. *ICCV*, 2011. 2, 4

[35] Kanglin Liu, Wenming Tang, Fei Zhou, and Guoping Qiu. Spectral regularization for combating mode collapse in gans. In *ICCV*, pages 6382–6390, 2019. 3

[36] Takeru Miyato, Toshiki Katoaka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018. 3, 4

[37] Takeru Miyato and Masanori Koyama. Cgans with projection discriminator. In *ICLR*, 2018. 4
