International Journal of Computer Vision
https://doi.org/10.1007/s11263-024-02270-w

Polynomial Implicit Neural Framework for Promoting Shape Awareness in Generative Models

Utkarsh Nath[1,3] · **Rajhans Singh**[2,3] · **Ankita Shukla**[4] · **Kuldeep Kulkarni**[5] · **Pavan Turaga**[2,3]

Received: 31 March 2024 / Accepted: 30 September 2024

Abstract

Polynomial functions have been employed to represent shape-related information in 2D and 3D computer vision, even from the very early days of the field. In this paper, we present a framework using polynomial-type basis functions to promote shape awareness in contemporary generative architectures. The benefits of using a learnable form of polynomial basis functions as drop-in modules into generative architectures are several—including promoting shape awareness, a noticeable disentanglement of shape from texture, and high quality generation. To enable the architectures to have a small number of parameters, we further use implicit neural representations (INR) as the base architecture. Most INR architectures rely on sinusoidal positional encoding, which accounts for high-frequency information in data. However, the finite encoding size restricts the model's representational power. Higher representational power is critically needed to transition from representing a single given image to effectively representing large and diverse datasets. Our approach addresses this gap by representing an image with a polynomial function and eliminates the need for positional encodings. Therefore, to achieve a progressively higher degree of polynomial representation, we use element-wise multiplications between features and affine-transformed coordinate locations after every ReLU layer. The proposed method is evaluated qualitatively and quantitatively on large datasets such as ImageNet. The proposed Poly-INR model performs comparably to state-of-the-art generative models without any convolution, normalization, or self-attention layers, and with significantly fewer trainable parameters. With substantially fewer training parameters and higher representative power, our approach paves the way for broader adoption of INR models for generative modeling tasks in complex domains. The code is publicly available at https://github.com/Rajhans0/Poly_INR.

Keywords Generative models · Polynomial functions · Shape awareness · Implicit models

1 Introduction

Implicit neural representation (INR) Mildenhall et al. (2021), Sitzmann et al. (2020) is a widely used approach for representing signals as a continuous function, where the continuous function is parameterized by a neural network. INRs offer significant flexibility and expressivity, even when employing a simple neural network architecture. For example, when applied to an image, an INR allows easy zooming in or out and sampling images at any desired resolution. This capability results in enhanced performance, particularly in tasks like super-resolution.

Communicated by Hongbo Fu.

✉ Utkarsh Nath
unath@asu.edu

Rajhans Singh
rsingh70@asu.edu

Ankita Shukla
ankitas@unr.edu

Kuldeep Kulkarni
kulkulka@adobe.com

Pavan Turaga
pturaga@asu.edu

[1] School of Computing and Augmented Intelligence, Arizona State University, Tempe, AZ, USA

[2] School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ, USA

[3] Geometric Media Lab, School of Arts, Media, and Engineering, Arizona State University, Tempe, AZ, USA

[4] Computer Science and Engineering, University of Nevada, Reno, NV, USA

[5] Adobe Research, Bangalore, Karnataka, India

Published online: 20 December 2024

INRs have demonstrated efficacy in 3D scene reconstruction and rendering from very few training images Mildenhall et al. (2021), Barron et al. (2022), Martin-Brualla et al. (2021), Yu et al. (2021). However, they are usually trained to represent a single given scene, signal, or image. Recently, INRs have been implemented as a generative model to generate entire image datasets leveraging various techniques such as Normalizing Flows Dupont et al. (2022a), GANs Chen and Zhang (2019), Skorokhodov et al. (2021), Anokhin et al. (2021), and Diffusion Models Dupont et al. (2022a), Zhuang et al. (2022). While they exhibit performance comparable to state-of-the-art generative models Rombach et al. (2022), Peebles and Xie (2023) on perfectly curated datasets like human faces Karras et al. (2019), their scalability to large, diverse datasets like ImageNet Deng et al. (2009) remains a challenge.

An INR typically comprises a positional encoding module and a multi-layer perceptron model (MLP). The positional encoding in INR commonly relies on sinusoidal functions, often referred to as Fourier features. Despite its widespread use, the capacity of such INR designs can be limiting for two key reasons. First, the size of the embedding space is limited; hence only a finite and fixed combination of periodic functions can be used, limiting its application to smaller datasets. Large diverse datasets inherently contain a variety of shapes, and current INR models struggle to represent complex geometries. Second, such an INR design needs to be mathematically coherent. These INR models can be interpreted as a non-linear combination of periodic functions where periodic functions define the initial part of the network, and the later part is often a ReLU-based non-linear function. Contrary to this, classical transforms (Fourier, sine, or cosine) represent an image by a linear summation of periodic functions. However, simply using a linear combination of the positional embedding in a neural network is also limiting, making it difficult to represent large and diverse datasets.

In this paper, we aim to construct an INR-based model capable of representing intricate shapes and generating large, diverse datasets such as ImageNet. We hypothesize that representing an image as a polynomial function of its coordinate location can effectively capture diverse structures in large datasets. Hence, rather than utilizing periodic functions, we propose Poly-INR, which models images as a polynomial function of their coordinate location. The main advantage of polynomial representation is the easy parameterization of polynomial coefficients with MLP. However, conventionally, MLP can only approximate lower-order polynomials. One can use a polynomial positional embedding of the form $x^p y^q$ in the first layer to enable the MLP to approximate higher order. However, such a design is limiting, as a fixed embedding size incorporates only fixed polynomial degrees. In addition, we do not know the importance of each polynomial degree beforehand for a given image. Hence, we do not use any positional encoding, but we progressively increase the degree of the polynomial with multiple levels of MLP.

Reconstructing image using polynomial bases comes from geometric moment literature. Rigid and non-rigid aspects of shape can be described in terms of geometric moments. Geometric moments have a long history in the vision community for a wide variety of applications ranging from invariant pattern recognition Hu (1962), Khotanzad and Hong (1990), Alajlan et al. (2008), Flusser and Suk (1993), segmentation Reeves et al. (1988), Foulonneau et al. (2006) and 3D shape recognition Tuceryan (1994), Sadjadi and Hall (1980), Elad and Kimmel (2003). Geometric moments are a very specific type of weighted averages of image pixel intensities, where the weights are drawn from specific polynomial-type basis functions. This operation can be expressed as a projection of the image on the bases. Despite classic theoretical developments around moments using specific choices of basis functions, their application to challenging tasks such as image classification and generation has remained limited. In this paper, we reintroduce geometric moments as a learn-

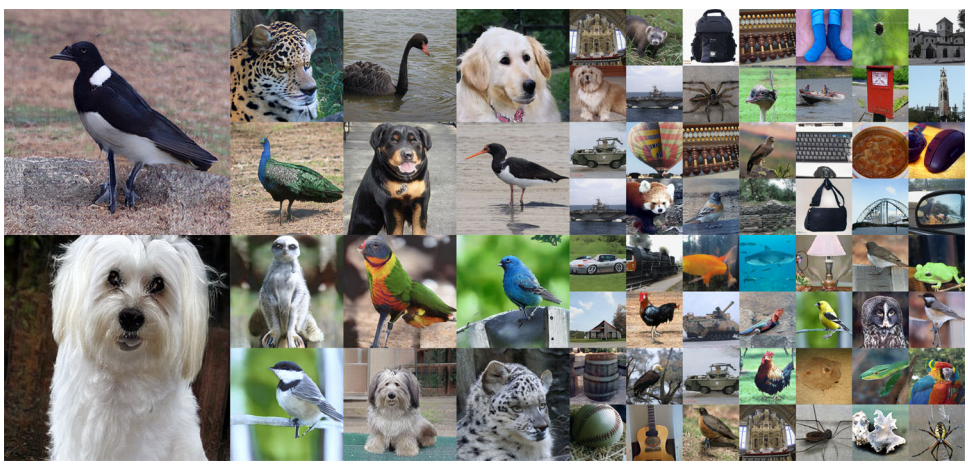

Fig. 1 Samples generated by our Poly-INR model on the ImageNet dataset at various resolutions. Our model generates images with high fidelity without using convolution, upsample, or self-attention layers, i.e., no interaction between the pixels

able spatial operation, presenting modules inspired by the image-projection analogy. We propose the Deep Geometric Moment (DGM) module as a fundamental building block in Poly-INR, designed to generate features that enhance shape awareness. Within the DGM framework, bases are learned in an end-to-end, differentiable manner. The DGM model conducts element-wise multiplication between the feature and affine transformed coordinate location, obtained after every ReLU layer. The affine parameters are parameterized by the latent code sampled from a known distribution. This way, our network learns the required polynomial order and represents complex datasets with considerably fewer trainable parameters.

In particular, the key highlights are summarized as follows:

- Poly-INR as a generative model performs comparably to the state-of-the-art CNN-based GAN model (StyleGAN-XL Sauer et al. (2022)) on the ImageNet dataset with $3-$ $-4\times$ fewer trainable parameters (depending on output resolution).
- Poly-INR outperforms the previously proposed INR models on the FFHQ dataset Karras et al. (2019), using a significantly smaller model.
- Poly-INR is extensively evaluated for interpolation, inversion, style-mixing, high-resolution sampling, and extrapolation.
- We demonstrate the applicability of Poly-INR in various applications such as multi-view image generation and text-to-image generation.

2 Related Work and Background

In this section, we discuss several INR-based generative models and recent advancements in the field of image generation. Additionally, we also provide an overview of geometric moment.

2.1 INR-Based Generative Models

Numerous studies have investigated the application of INR in generative modeling, capitalizing on its continuous nature and expressive capabilities. For example, CIPS Anokhin et al. (2021) uses Fourier features and learnable vectors for each spatial location as positional encoding and uses StyleGAN-like weight modulation for layers in the MLP. Similarly, INR-GAN Skorokhodov et al. (2021) proposes a multi-scale generator model where a hyper-network determines the parameters of the MLP. INR-GAN has been further extended to generate an 'infinite'-size continuous image using anchors Skorokhodov et al. (2021). Zhuang et al. (2022) employs diffusion models to explicit signal fields to generate samples at different modalities such as 2D images and 3D geometry. Meanwhile, others Dupont et al. (2022b), Dupont et al. (2022a), Du et al. (2021) try to model the weight distribution of INRs with GANs or diffusion models. However, these INR-based models encounter scalability challenges when applied to large-scale datasets and have only demonstrated promising results on smaller datasets. Our work scales easily to large datasets like ImageNet (Table 1) owing to the significantly fewer parameters.

Other approaches have combined CNNs with coordinate-based features. For example, the Local Implicit Image Function (LIIF) Chen et al. (2021) and Spherical Local Implicit Image Function (SLIIF) Yoon et al. (2022) use a CNN-based backbone to generate feature vectors corresponding to each coordinate location. Arbitrary-scale image synthesis Ntavelis et al. (2022) uses a multi-scale convolution-based generator model with scale-aware position embedding to generate scale-consistent images. StyleGAN model, further extended by Karras et al. (2021) (StyleGAN-3) to use coordinate location-based Fourier features. In addition, StyleGAN-3 uses filter kernels equivariant to the coordinate grid's translation and rotation. However, the rotation equivariant version of the StyleGAN-3 model fails to scale to the ImageNet dataset, as reported in Sauer et al. (2022). Instead of using convolution layers, the Poly-INR only uses linear and ReLU layers.

2.2 Image Generation

Diffusion models Ho et al. (2020), Ho et al. (2022), Song et al. (2021) have shown notable success in image generation tasks, consistently delivering high-quality results and broad distribution coverage Dhariwal and Nichol (2021), often outperforming GANs. However, their iterative reverse process, which typically involves a large number of steps (e.g., 1000 steps), makes them significantly slower and less efficient compared to GANs. In this paper, our INR model is built upon GANs. GANs have been widely used for image generation and synthesis tasks Goodfellow et al. (2020). In recent work, several improvements have been proposed Radford et al. (2016), Karras et al. (2019), Miyato et al. (2018), Arjovsky et al. (2017), Gulrajani et al. (2017) over the original architecture. For example, the popularly used StyleGAN Karras et al. (2019) model uses a mapping network to generate style codes, which are then used to modulate the weights of the Conv layers. StyleGAN improves image fidelity, as well as enhances inversion and image editing capabilities Härkönen et al. (2020). StyleGAN has been scaled to large datasets like ImageNet Sauer et al. (2022), using a discriminator that uses projected features from a pre-trained classifier Sauer et al. (2021). More recently, transformer-based models have also been used as generators Zhao et al. (2021), Lee et al. (2021), Gao et al. (2023); however, the self-attention mechanism is computationally costly for achieving higher resolution.

Unlike these methods, our generator is free of convolution, normalization, and self-attention mechanisms and only uses ReLU and Linear layers to achieve competitive results, but with far fewer parameters (Tables 1 and 2). Concurrently, recent advancements in generative models have introduced patch-wise training Skorokhodov et al. (2024), Ding et al. (2023), Zheng et al. (2023), Arakawa et al. (2023), Wang et al. (2024), Wang et al. (2022), Skorokhodov et al. (2022), a method where images are divided into smaller patches that are processed independently to enhance training efficiency and effectively manage memory usage. Additionally, Zheng et al. (2023) incorporated neural operators for more efficient training and evaluations. In contrast, the Poly-INR approach processes the entire image at once but generates each pixel independently, rather than generating image patches, and thus offers a fundamentally different methodology for image synthesis. However, Poly-INR can also adopt patch-based methods to expedite training and reduce memory demands, providing adaptability for memory-constrained devices.

In recent years, there has been increasing interest in a new class of architectures termed deep polynomial neural networks Chrysos et al. (2022), Wu et al. (2022). The goal of these approaches is to specifically model the input–output relationship of a neural network using a polynomial function defined in the high-dimensional spaces matched to the input and output dimensions. Our approach may be considered related to these approaches but is more specific to 2D polynomial functions, defined on image-grids, which allow us to interpret the learned polynomials specifically as promoting shape-awareness. This interpretation is supported by the prior work in using similar type of polynomials for 2D (and 3D) shape representations Hu (1962).

2.3 Geometric Moment

A *moment* for a given two-dimensional piece-wise continuous function $f(x, y)$ is defined as:

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q f(x, y)dxdy, \tag{1}$$

where, (x, y) is the 2D coordinate and $(p+q)$ is the order of the moment. By uniqueness theorem Hu (1962), if $f(x, y)$ is a piece-wise bounded continuous function (i.e. it is non-zero only on a compact part of the xy plane), then the moment sequence m_{pq} is uniquely defined for all orders $(p + q)$ by $f(x, y)$. Conversely, $f(x, y)$ is uniquely determined by the sequence m_{pq}. The geometric moment for a discrete 2D function $I(x, y)$ is given by the discrete version of (1):

$$m_{pq} = \sum_x \sum_y x^p y^q I(x, y) \tag{2}$$

where, x, y are the 2D coordinates and $(p + q)$ is the order of the moment. Equivalently, moments can also be seen as a 'projection' of the 2D function on certain bases of the form $x^p y^q$. Instead of using the bases function of type $x^p y^q$, one can instead also use orthogonal functions like Legendre or Zernike polynomials Teague (1980) for better reconstruction. Image moments are well-known invariant shape descriptors with a long history of use in the computer vision literature to capture the geometrical properties of an image. For example, m_{00} (0^{th} order) represents average pixel intensity, m_{10} (1^{st} order) and m_{01} (1^{st} order) represent xy centroid coordinate and the combination of 1^{st} and 2^{nd} order can be used to compute orientation.

An early work by Hu (1962) introduced a way to find invariant moments for images. The Hu moments consist of seven moments, mostly a combination of lower-order moments invariant under scaling, translation, and rotation. While these basic sets of seven Hu moments are provably invariant to rotation, translation, and scale, their use has been limited since their discriminative power is not very high. Developing invariant moments for the Legendre and Zernike polynomials for any arbitrary order is also possible Chong et al. (2004), Zhang et al. (2011), Zhang et al. (2009), Khotanzad and Hong (1990), Kim and Lee (2003), Wang and Healey (1998), Yap and Paramesran (2005), Flusser et al. (2003). However, they also have not significantly impacted contemporary image classification or generative tasks.

In this paper, we seek to advance a new approach for defining spatial operations for image generation and classification networks, whose structure is motivated by classic moment computation but whose basis functions are left to be learned end-to-end by a deep learning network in a task-specific way. This implies that we are not seeking to replicate any of the classical moments in an exact sense but to find ways to fuse moment-like computations and let networks learn the suitable basis functions for a given task. This approach is described in the next section.

2.3.1 Geometric Moments and Deep Networks

There has been prior work in integrating geometric moments with deep networks, as specifically applied to 3D shape classification, from point-cloud data. For example, geodesic moment-based features from an auto-encoder were used to classify 3D shapes Luciano and Hamza (2018). On the other hand, CNNs were used as a polynomial function to learn bases and the needed affine transformation parameters for 3D point cloud data-based shape classification Joseph-Rivlin et al. (2019). This line of work was extended in Li et al. (2020), which uses graph CNN to capture local features of the 3D object. More recently, Theodoridis et al. (2021) and Wu et al. (2017) replaced the conventional global average pooling in CNN models with invariant Zernike moment-based pooling

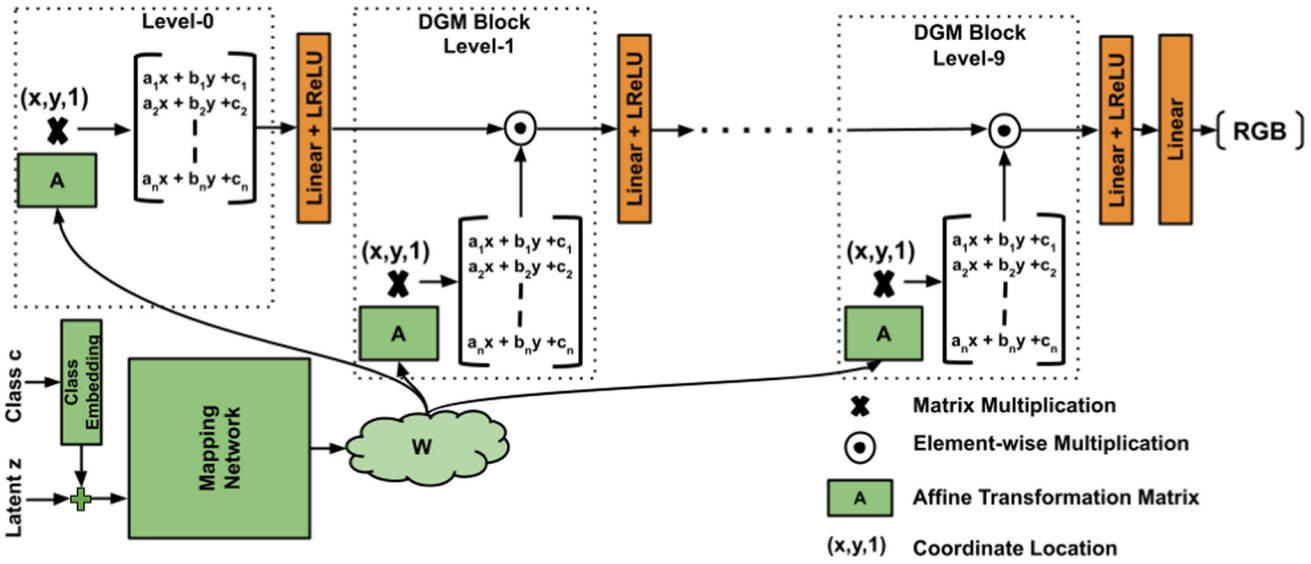

Fig. 2 Overview of our proposed Polynomial Implicit Neural Representation (Poly-INR) based generator architecture. Our model consists of two networks: (1) Mapping network, which generates the affine parameters from the latent code z, and (2) Synthesis network, which synthesizes the RGB value for the given pixel location. Our Poly-INR model is defined using only Linear and ReLU layers end-to-end

for image classification tasks. In contrast to these approaches, our method is specifically tailored for generative tasks, where we differentially learn bases and affine parameters.

3 Poly-INR

We are interested in a class of functions that represent an image in the form:

$$G(x, y) = g_{00} + g_{10}x + g_{01}y + ... + g_{pq}x^p y^q, \tag{3}$$

where (x, y) is the normalized pixel location sampled from a coordinate grid of size $(H \times W)$, while the coefficients of the polynomial (g_{pq}) are parameterized by a latent vector z sampled from a known distribution and are independent of the pixel location. Therefore, to form an image, we evaluate the generator G for all pixel locations (x, y) for a given fixed z:

$$I = \{G(x, y; z) \,|\, (x, y) \in Grid(H, W)\}, \tag{4}$$

where, $Grid(H, W) = \{(\frac{x}{W-1}, \frac{y}{H-1}) \,|\, 0 \leq x < W, 0 \leq y < H\}$. By sampling different latent vectors z, we generate different polynomials and represent images over a distribution of real images.

Our goal is to learn the polynomial defined by (3) using only Linear and ReLU layers to represent diverse datasets inherently containing a variety of shapes and geometries. However, the conventional definition of MLP usually takes the coordinate location as input, processed by a few Linear and ReLU layers. This definition of INR can only approximate low-order polynomials and hence only generates low-frequency information. Although, one can use a positional embedding consisting of polynomials of the form $x^p y^q$ to approximate a higher-order polynomial. However, this definition of INR is limiting since a fixed-size embedding space can contain only a small combination of polynomial orders. Furthermore, we do not know which polynomial order is essential to generate the image beforehand. Hence, we progressively increase the polynomial order in the network and let it learn the required orders. We implement this using the DGM module. Within the DGM module, we conduct element-wise multiplication of features with the affine-transformed coordinate location at different levels, shown in Fig. 2. Our model consists of two parts: 1) **Mapping network**, which takes the latent code z and maps it to affine parameters space $\mathbf{W}$, and 2) **Synthesis network**, which takes the pixel location and generates the corresponding RGB value.

3.1 Mapping Network

The mapping network takes the latent code $z \in \mathbb{R}^{64}$ and maps it to the space $\mathbf{W} \in \mathbb{R}^{512}$. Our model adopts the mapping network used in Sauer et al. (2022). It consists of a pre-trained class embedding, which embeds the one hot class label into a 512 dimension vector and concatenates it with the latent code z. Then, the mapping network consists of an MLP with two layers, which maps it to the space $\mathbf{W}$. We use this $\mathbf{W}$ to

generate affine parameters by using additional linear layers; hence, we refer to the set of **W**s as affine parameters space.

3.2 Synthesis Network

The synthesis network generates the RGB ($\mathbb{R}^3$) value for the given pixel location (x, y). We are interested in reconstructing images with a diverse set of shapes and structures. Geometric moments capture the shape information exceptionally well and provide discriminative cues; however, their discrimination power is quite limited and generally requires a salient object over a homogeneous background. To address this limitation, we blend the strength of geometric moment and neural networks. We propose Deep Geometric Moment (DGM) that uses geometric moments along with neural networks to provide both shape and texture features.

In the traditional usage of moments in vision, the number and the order of moments is an experimental design choice. Choosing the right number and order of moments depends on the underlying tasks; large moments are useful for image reconstruction, whereas, for image classification, higher-order moments are affected by noise and, hence, not very useful. Thus, selecting the correct moment order is essential. In our method, we specify the required number of moments (in terms of feature dimension), but the exact basis functions and orders are learned by the networks end-to-end.

The DGM module is defined by (5), we project the relevant features $f(x, y)$ into the learned coordinate bases:

$$DGM(A, g, f) = (A \times g(x, y)) \odot f(x, y), \tag{5}$$

where, $g(x, y)$ is a learnable 2D polynomial function, $f(x, y)$ is image feature at coordinate location (x, y), A refers to the affine transformation parameters and $\odot$ is element-wise multiplication. Affine transformations include four basic transformations: translation, scaling, rotation, and shear. For a 2D point represented by coordinates x, y, its affine transformation can be described by $ax+by+c$. Here, a and b are parameters influencing the transformation through scaling, rotation, or shearing effects, while c typically represents translation, allowing modification of the point within the 2D space. To account for varying locations, sizes, poses, and deformation, we allow our network to learn affine parameters to appropriately deform the 2D coordinate grid during moment computation (as shown in Fig. 2).

The synthesis network consists of multiple levels; at each level, it receives the affine transformation parameters from the mapping network and the pixel coordinate location. At *level*-0, we affine transform the coordinate grid and feed it to a Linear layer followed by a Leaky-ReLU layer with $negative\_slope = 0.2$. At later levels, we use the DGM module to conduct element-wise multiplication between the feature from the previous level and the affine-transformed coordinate grid and then feed it to Linear and Leaky-ReLU layers. With the element-wise multiplication at each level, the network retains the flexibility to either elevate the order of the x or y coordinate positions or maintain the current order by setting the corresponding affine transformation coefficients $a = b = 0$ and $c = 1$. In our model, we use 10 levels, which is sufficient to generate large datasets like ImageNet. Mathematically, the synthesis network can be expressed as follows:

$$F_i = \begin{cases} \sigma(W_0(A_0 \times X)), & i = 0 \\ \sigma(W_i \times DGM_i(A_i, X, F_{i-1}), & i > 0 \end{cases} \tag{6}$$

where F_i represents the output at *level-i*, $X \in \mathbb{R}^{3\times HW}$ is the coordinate grid of size $H \times W$ with an additional dimension for the bias, $A_i \in \mathbb{R}^{n\times 3}$ is the affine transformation matrix from the mapping network for *level-i*, $W_i \in \mathbb{R}^{n\times n}$ is the weight of the linear layer at *level-i* and σ is the Leaky-ReLU layer. Here, n is the dimension of the feature channel in the synthesis network, which is the same for all levels. For large datasets like ImageNet, we choose the channel dimension $n = 1024$, and for smaller datasets like FFHQ, we choose $n = 512$. Note that with this definition, our model only uses Linear and ReLU layers end-to-end and synthesizes each pixel independently.

Relation to StyleGAN: StyleGANs Karras et al. (2019), Karras et al. (2020), Karras et al. (2021) can be seen as a special case of our formulation. By keeping the coefficients (a_j, b_j) in the affine transformation matrix of x and y coordinate location equal to zero, the bias term c_j would act as a style code. However, our affine transformation adds location bias to the style code rather than just using the same style code for all locations in StyleGAN models. This location bias makes the model very flexible in applying a style code only to a specific image region, making it more expressive. In addition, our model differs from the StyleGANs in many aspects. First, our method does not use weight modulation/demodulation or normalizing Karras et al. (2020) tricks. Second, our model does not employ low-pass filters or convolutional layers. Finally, we do not inject any spatial noise into our synthesis network. We can also use these tricks to improve the model's performance further. However, our model's definition is straightforward compared to other GAN models.

4 Experiments

The effectiveness of our model is evaluated on two datasets: 1) ImageNet Deng et al. (2009) and 2) FFHQ Karras et al. (2019). The ImageNet dataset consists of $1.2M$ images over $1K$ classes, whereas the FFHQ dataset contains $\sim 70K$ images of curated human faces. All our models have 64 dimensional latent space sampled from a normal distribution

Table 1 Quantitative comparison of Poly-INR method with CNN-based generative models on ImageNet datasets

(a) ImageNet 128 × 128

Model	FID ↓	sFID ↓	rFID ↓	IS ↑	Pr ↑	Rec ↑
BigGAN	6.02	7.18	6.09	145.83	0.86	0.35
CDM	3.52	–	-	128.80	–	–
ADM	5.91	5.09	13.29	93.31	0.70	0.65
ADM-G	2.97	5.09	3.80	141.37	0.78	0.59
StyleGAN-XL	1.81	3.82	1.82	200.55	0.77	0.55
Poly-INR	2.08	3.93	2.76	179.64	0.70	0.45

(b) ImageNet 256 × 256

Model	FID ↓	sFID ↓	rFID ↓	IS ↑	Pr ↑	Rec ↑
BigGAN	6.95	7.36	75.24	202.65	0.87	0.28
ADM	10.94	6.02	125.78	100.98	0.69	0.63
ADM-G	3.94	6.14	11.86	215.84	0.83	0.53
DiT-XL/2-G	2.27	4.60	–	278.54	0.83	0.57
StyleGAN-XL	2.30	4.02	7.06	265.12	0.78	0.53
Poly-INR	2.86	4.37	7.79	241.43	0.71	0.39

(c) ImageNet 512 × 512

Model	FID ↓	sFID ↓	rFID ↓	IS ↑	Pr ↑	Rec ↑
BigGAN	8.43	8.13	312.00	177.90	0.88	0.29
ADM	23.24	10.19	561.32	58.06	0.73	0.60
ADM-G	3.85	5.86	210.83	221.72	0.84	0.53
DiT-XL/2-G	3.04	5.04	–	240.82	0.84	0.54
StyleGAN-XL	2.41	4.06	51.54	267.75	0.77	0.52
Poly-INR	3.81	5.06	54.31	267.44	0.70	0.34

(d) Number of parameters in millions (M)

Model	64^2	128^2	256^2	512^2
BigGAN	-	141.0	164.3	164.7
ADM	296.0	422.0	554.0	559.0
DiT-XL	-	-	675.0	675.0
StyleGAN-XL	134.4	158.7	166.3	168.4
Poly-INR	46.0	46.0	46.0	46.0

(d) compares the number of parameters used in all models at various resolutions. The results for existing methods are quoted from the StyleGAN-XL paper

Table 2 Quantitative comparison of Poly-INR method with CNN and INR-based generative models on FFHQ dataset at 256 × 256

Model	# Params (M)	FID ↓	Inference time (sec/img)
StyleGAN2	30.0	3.83	0.016
StyleGAN-XL	67.9	2.19	0.047
CIPS	45.9	4.38	0.067
INR-GAN	72.4	4.95	0.024
Poly-INR	13.6	2.72	0.054

with mean 0 and standard deviation 1. The affine parameters space $\mathbf{W}$ of the mapping network is 512 dimensions, and the synthesis network consists of 10 levels with feature dimension $n = 1024$ for the ImageNet and $n = 512$ for FFHQ. We follow the training scheme of the StyleGAN-XL method Sauer et al. (2022) and use a projected discriminator based on the pre-trained classifiers (DeiT Touvron et al. (2021), and EfficientNet Tan and Le (2019)) with an additional classifier guidance loss Dhariwal and Nichol (2021).

We train our model progressively with increasing resolution, i.e., we start by training at low resolution and continue training with higher resolutions as training progresses. Since the computational cost is less at low resolution, the model is trained for a large number of iterations, followed by training for high resolution. Since the model is already trained at low resolution, fewer iterations are needed for convergence at high resolution. However, unlike StyleGAN-XL, which freezes the previously trained layers and introduces new layers for higher resolution, Poly-INR uses a fixed number of layers and trains all the parameters at every resolution.

4.1 Quantitative Results

We compare our model against CNN-based GANs (BigGAN Brock et al. (2018) and StyleGAN-XL Sauer et al. (2022)) and diffusion models (CDM Ho et al. (2022), ADM, ADM-G Dhariwal and Nichol (2021), and DiT-XL Peebles and Xie (2023)) on the ImageNet dataset. We also report results on the FFHQ dataset for INR-based GANs (CIPS Anokhin et al. (2021) and INR-GAN Skorokhodov et al. (2021)) as they do not train models on ImageNet.

Quantitative metrics: We use Inception Score (IS) Salimans et al. (2016), Frechet Inception Distance (FID) Heusel et al. (2017), Spatial Frechet Inception Distance (sFID) Nash et al. (2021), random-FID (rFID) Sauer et al. (2022), precision (Pr), and recall (Rec) Kynkäänniemi et al. (2019). IS (higher the better) quantifies the quality and diversity of the generated samples based on the predicted label distribution by the Inception network but does not compare the distribution of the generated samples with the real distribution. The FID score (lower the better) overcomes this drawback by measuring the Frechet distance between the generated and real distribution in the Inception feature space. Further, sFID uses higher spatial features from the Inception network to account for the spatial structure of the generated image. Like StyleGAN-XL, we also use the rFID score to ensure that the network is not just optimizing for IS and FID scores. We use the same randomly initialized Inception network provided by Sauer et al. (2022). In addition, we also compare our model on the precision and recall metric (higher the better), which measures how likely the generated sample is from the real distribution.

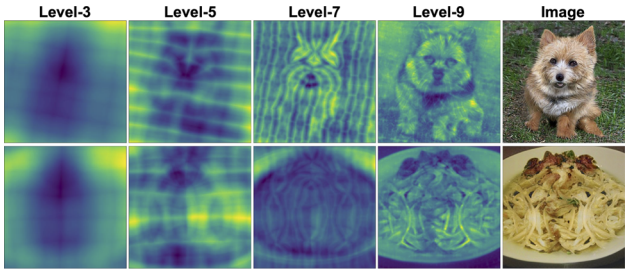

Fig. 3 Heatmap visualization at different levels of the synthesis network. At initial levels, the model captures the basic shape of the object, and at higher levels, the image's finer details are captured

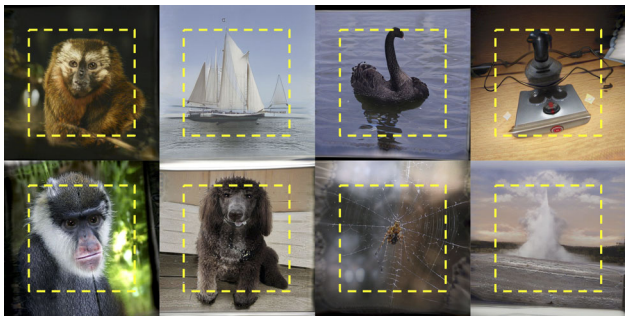

Fig. 4 Few example images showing extrapolation outside the image boundary (yellow square). The Poly-INR model is trained to generate images on the coordinate grid $[0, 1]^2$. For extrapolation, we use the grid size $[-0.25, 1.25]^2$. Our model generates continuous image outside the conventional boundary (Color figure online)

Table 1 summarizes the results on the ImageNet dataset at different resolutions. The results for existing methods are quoted from the StyleGAN-XL paper. We observe that the performance of the proposed model is third best after DiT-XL and StyleGAN-XL on the FID and IS metrics. The proposed model outperforms the ADM and BigGAN models at all resolutions and performs comparably to the StyleGAN-XL at 128×128 and 256×256. We also observe that with the increase in image size, the FID score for Poly-INR drops much more than StyleGAN-XL. The FID score drops more because our model does not add any additional layers with the increase in image size. For example, the StyleGAN-XL uses 134.4M parameters at 64×64 and 168.4M at 512×512, whereas Poly-INR uses only 46.0M parameters at every resolution, as reported in Table 1d. The table shows that our model performs comparably to the state-of-the-art CNN-based generative models, even with significantly fewer parameters. On precision metric, the Poly-INR method performs comparably to other methods; however, the recall value is slightly lower compared to StyleGAN-XL and diffusion models at higher resolution. Again, this is due to the small model size, limiting the model's capacity to represent much finer details at a higher resolution.

We also compare the proposed method with other INR-based GANs, such as CIPS and INR-GAN, on the FFHQ

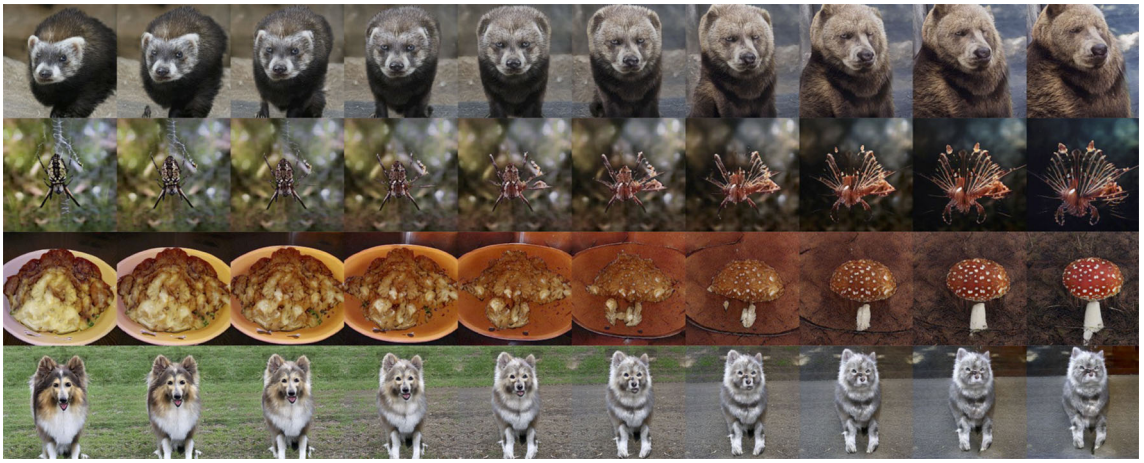

Fig. 5 Linear interpolation between two random points. The first two rows represent interpolation in the latent space, while in the last two, we directly interpolate between the affine parameters. Poly-INR provides smooth interpolation even in a high dimension of affine parameters. Our model generates high-fidelity images similar to state-of-the-art models like StyleGAN-XL but without the need for convolution or a self-attention mechanism. Comparisons with existing methods are present in the supplementary material

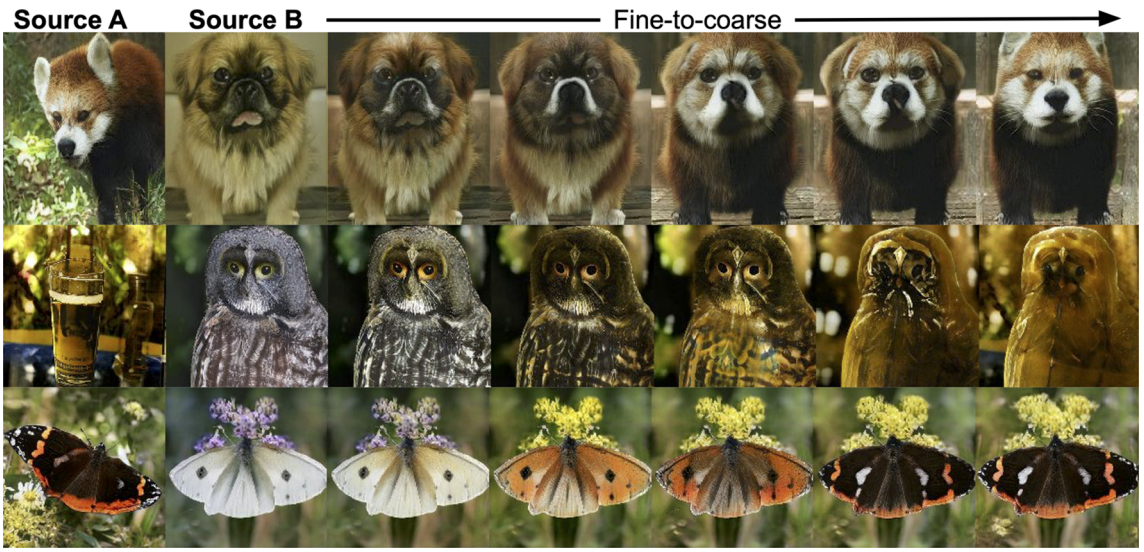

Fig. 6 Source A and B images are generated corresponding to random latent codes, and the rest of the images are generated by copying the affine parameters of source A to source B at different levels. Copying the higher levels' (8 and 9) affine parameters leads to finer style changes, whereas copying the middle levels' (7, 6, and 5) leads to coarse style changes

dataset. Table 2 shows that the proposed model significantly outperforms these models, even with a small generator model. Interestingly, the Poly-INR method outperforms the StyleGAN-2 and performs similarly to StyleGAN-XL, using significantly fewer parameters. Table 2 also reports the inference speed of these models on a Nvidia-RTX-6000 GPU. StyleGANs and INR-GAN employ a multi-scale architectural approach, initiating the image synthesis process with a compact latent representation and subsequently upscaling it progressively to achieve the desired resolution. In contrast, CIPS and Poly-INR models synthesize each pixel independently, with all computations performed at the same resolution as the output image, thus leading to increased inference time.

4.2 Qualitative Results

Figure 1 shows images sampled at different resolutions by the Poly-INR model trained on 512×512. We observe that our model generates diverse images with very high fidelity. Even though the model does not use convolution or self-attention layers, it generates realistic images over datasets

Fig. 7 The Poly-INR model generates smooth interpolation with embedded images in affine parameters space. The leftmost image (first row) is from the ImageNet validation set, and the last two (rightmost) are the OOD images

like ImageNet. In addition, the model provides flexibility to generate images at different scales by changing the size of the coordinate grid, making the model efficient if low-resolution images are needed for a downstream task. In contrast, CNN-based models generate images only at the training resolution due to the non-equivariant nature of the convolution kernels to image scale.

4.2.1 Heatmap Visualization

Figure 3 visualizes the heatmap at different levels of our synthesis network. To visualize a feature as a heatmap, we first compute the mean along the spatial dimension of the feature and use it as a weight to sum the feature along the channel dimension. In the figure, we observe that in the initial levels (0–3), the model forms the basic structure of the object. Meanwhile, in the middle levels (4–6), it captures the object's overall shape, and in the higher levels (7–9), it adds finer details about the object. Furthermore, we can interpret this observation in terms of polynomial order. Initially, it only approximates low-order polynomials and represents only basic shapes. However, at higher levels, it approximates higher-order polynomials representing finer details of the image.

4.2.2 Extrapolation

The INR model is a continuous function of the coordinate location; hence, we extrapolate the image by feeding the pixel location outside the conventional image boundary. Our Poly-INR model is trained to generate images on the coordinate grid defined by $[0, 1]^2$. We feed the grid size $[-0.25, 1.25]^2$ to the synthesis network to generate the extrapolated images. Figure 4 shows a few examples of extrapolated images. In the figure, the region within the yellow square represents the conventional coordinate grid $[0, 1]^2$. The figure shows that our INR model not only generates a continuous image outside

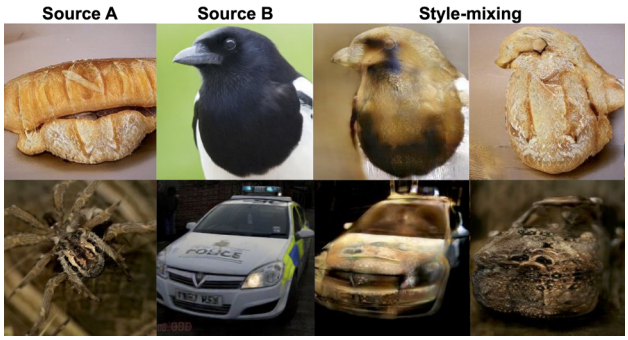

Fig. 8 Style-mixing with embedded images in affine parameters space. Source B is the embedded image from the ImageNet validation set, mixed with the style of randomly sampled source A image

the boundary but also preserves the geometry of the object present within the yellow square. However, in some cases, the model generates a black or white image border, resulting from the image border present in some real images of the training set.

4.2.3 Sampling at Higher-Resolution

Another advantage of using our model is the flexibility to generate images at any resolution, even if the model is trained on a lower resolution. We generate a higher-resolution image by sampling a dense coordinate grid within the $[0, 1]^2$ range. Table 3 shows the FID score evaluated at 512×512 for models trained on the lower-resolution ImageNet dataset. We compare the quality of upsampled images generated by our model against the classical interpolation-based upsampling methods. The table shows that our model generates crisper upsampled images, achieving a significantly better FID score than the classical interpolation-based upsampling method. However, we do not observe significant FID score improvement for our Poly-INR model trained on 128×128 or higher resolution against the classical interpolation techniques. This could be due to the limitations of the ImageNet

Table 3 FID score (lower the better) evaluated at 512 × 512 for models trained at a lower resolution and compared against classical interpolation-based upsampling

Training Resolution	Nearest Neighbour	Bilinear	Bicubic	Poly-INR
32 × 32	184.39	112.28	73.86	65.15
64 × 64	89.24	72.41	42.97	36.30

We conduct a qualitative comparison of images generated by all methods in Sect. A.7 of the Appendix

dataset, which primarily consists of lower-resolution images than the 512 × 512. We used bilinear interpolation to prepare the training dataset at 512 × 512. We believe this performance can be improved when the model has access to higher-resolution images for training. We also compare the upsampling performance with other INR-based GANs by reporting the FID scores at 1024 × 1024 for models trained on FFHQ-256 × 256 as follows: **Poly-INR:13.69, INR-GAN: 18.51, CIPS:29.59**. Our Poly-INR model provides better high-resolution sampling than the other two INR-based generators.

4.2.4 Interpolation

Figure 5 shows that our model generates smooth interpolation between two randomly sampled images. In the first two rows of the figure, we interpolate in the latent space, and in the last two rows, we directly interpolate between the affine parameters. In our synthesis network, only the affine parameters depend on the image, and other parameters are fixed for every image. Hence interpolating in affine parameter space means interpolation in INR space. Our model provides smoother interpolation even in the affine parameters space and interpolates with the geometrically coherent movement of different object parts. For example, in the first row, the eyes, nose, and mouth move systematically with the whole face.

4.2.5 Style-Mixing

Similar to StyleGANs, our Poly-INR model transfers the style of one image to another. Our model generates smooth style mixing even though we do not use any style-mixing regularization during the training. Figure 6 shows examples of style-mixing from source A to source B images. For style mixing, we first obtain the affine parameters corresponding to the source A and B images and then copy the affine parameters of A to B at various levels of the synthesis network. Copying affine parameters to higher levels (8 and 9) leads to finer changes in the style, while copying to middle levels (7, 6, and 5) leads to the coarse style change. Mixing the affine parameters at initial levels changes the shape of the generated object. In the figure, we observe that our model provides smooth style mixing while preserving the original shape of the source B object.

4.2.6 Inversion

Embedding a given image into the latent space of the GAN is an essential step for image manipulation. In our Poly-INR model, for inversion, we optimize the affine parameters to minimize the reconstruction loss, keeping the synthesis network's parameters fixed. We use VGG feature-based perceptual loss for optimization. We embed the ImageNet validation set in the affine parameters space for the quantitative evaluation. Our Poly-INR method effectively embeds images with high PSNR scores (**PSNR:26.52** and **SSIM:0.76**), outperforming StyleGAN-XL (PSNR: 13.5 and SSIM: 0.33). However, our affine parameters dimension is much larger than the StyleGAN-XL's latent space. Even though the dimension of the affine parameters is much higher, the Poly-INR model provides smooth interpolation for the embedded image. Figure 7 shows examples of interpolation with embedded images. In the figure, the first row (leftmost) is the embedded image from the validation set, and the last two rows (rightmost) are the out-of-distribution images. Surprisingly, our model provides smooth interpolation for OOD images. In addition, Fig. 8 shows smooth style-mixing with the embedded images. In some cases, we observe that the fidelity of the interpolated or style-mixed image with the embedded image is slightly lower compared to samples from the training distribution. This is due to the large dimension of the embedding space, which sometimes makes the embedded point farther from the training distribution. It is possible to improve interpolation quality further by using the recently proposed pivotal tuning inversion method Roich et al. (2022), which finetunes the generator's parameters around the embedded point.

4.3 Multi-View Consistent Images

In this section, we leverage PolyINR to generate multi-view-consistent images and 3D geometries in real-time. We employ the EG3D Chan et al. (2022), a pipeline introduced for unsupervised 3D representation learning from single-view 2D images. EG3D adopts a tri-plane formulation, aligning explicit features along three axis-aligned orthogonal feature planes, each characterized by a spatial resolution of $N \times N \times C$, where N represents spatial resolution and C denotes the number of channels.

To generate a 3D point $x \in \mathbb{R}^3$, EG3D projects it onto each of the three feature planes, retrieving the corresponding fea-

Fig. 9 Multi-view images generated at 128 × 128 synthesized by PolyINR model trained on FFHQ dataset. We use PTI Roich et al. (2022) to fit a target image generated from PolyINR trained along with the EG3D pipeline and recover the underlying 3D shape

ture vectors (F_{xy}, F_{yz}, F_{xz}) through bilinear interpolation, and aggregates these vectors via summation. Subsequently, a lightweight decoder network, implemented as a small MLP, interprets the aggregated 3D features F to derive color and density. These quantities are then rendered into RGB images using (neural) volume rendering Mildenhall et al. (2021) techniques.

The tri-plane representation facilitates efficient image rendering via neural volume rendering. In this section, we generate tri-plane features using Poly-INR backbone. We set N and C to 64 and 32. The neural renderer aggregates features from each of the 32-channel tri-planes and predicts 32-channel feature images from a given camera pose. Following Chan et al. (2022), a "super-resolution" module is employed to upsample and refine these raw neurally rendered images. The entire pipeline is trained end-to-end from random initialization, akin to the training scheme in StyleGAN2 Karras et al. (2020). Figure 9 represents images generated from Poly-INR as the generator along with the EG3D pipeline. We train our model on the FFHQ dataset at 128 × 128 resolution. In Table 4, we compare the model trained with the Poly-INR and StyleGAN2 within the EG3D pipeline. StyleGAN2 achieves a better FID score; however, it possesses 50% more parameters than Poly-INR.

Table 4 Quantitative comparison of Poly-INR for the multi-view image generation task on the FFHQ dataset at 128 × 128 resolution

Generator	FID ↓	# Params (M)
StyleGAN2	6.3	26.5
Poly-INR	8.5	14.3

4.4 Text-Guided Image Manipulation

In this section, we explore leveraging the power of Contrastive Language-Image Pre-training (CLIP) Radford et al. (2021) models to construct a text-based interface for manip-

Table 5 Ablation study on the number of levels used during Poly-INR training on ImageNet-100 at different resolutions

Levels	32^2	64^2	128^2	# Params
2	129.3	151.4	193.5	7.9
4	10.3	16.2	22.3	17.4
6	6.6	8.5	12.1	26.8
8	5.9	7.4	9.7	36.3
10	5.7	6.6	8.1	45.8

32^2, 64^2 and 128^2 represent FID scores corresponding to various number of levels at resolutions 32×32, 64×64 and 128×128 respectively. # Params represents number of parameters in millions. The channel dimension for the latent vector (z) is set to 1024

ulating the PolyINR model without manual intervention. Following StyleCLIP Patashnik et al. (2021), we adopt a training approach that employs a CLIP-based loss to optimize an input latent vector in response to user-provided text prompts. Let w be the input latent vector and t be the input text prompt. Then we optimise w using (7).

$$L_{T2I} = D_{CLIP}(G_{PolyINR}(w), t) + L_{ID}(w), \tag{7}$$

where $G_{PolyINR}$ represents pre-trained PolyINR generator. The cosine distance between the CLIP embeddings of its two arguments is denoted by D_{CLIP}. The similarity to the input image is regulated by the ℓ^2-norm in latent space and the identity loss Richardson et al. (2021), defined as: $L_{ID}(w) = 1 - \langle F(G(ws)); F(G(w)) \rangle$. Here, F represents a pretrained ArcFace network Deng et al. (2019) for face recognition, and $\langle \cdot, \cdot \rangle$ computes the cosine similarity between its arguments. We adhere to the same hyper-parameter settings as those established in StyleCLIP.

Figure 10 illustrates several examples generated by optimizing the latent input vector to PolyINR models based on various text prompts. On the left side of the figure, the transition of the generated image corresponding to each text prompt is displayed at several different iterations. For instance, when the input text prompt is "An elderly man with a beard," we observe the gradual transformation of the initially generated face from that of a young man to that of an older man with a growing beard.

4.5 Ablation Study

In this section, we conduct ablation studies to understand the significance of the number of levels, dimension of latent vector (z), and affine transformation in the PolyINR model. We conduct our experiments on ImageNet-100 Tian et al. (2020) at 32×32, 64×64 and 128×128 resolution. ImageNet-100 is a subset of ImageNet-1k dataset Russakovsky et al. (2015) with 100 classes and about $130k$ images. In Table 5 and 6, we observe that both increasing levels or latent dimensions improve the model's performance. Adding more levels or increasing latent dimension enhances the model's capacity. However, the latent dimension in Poly-INR has a more significant impact on performance than the number of levels. We observe that Poly-INR with ten levels and z set to 768 performs poorly against Poly-INR with six levels and z set to 1024, despite both models exhibiting a similar number of parameters. Furthermore, we find that decreasing z makes the training unstable. In Table 7, we conduct an ablation study to assess the impact of applying affine transformations at each level of the Poly-INR model. We observe that the removal of affine transformations from Poly-INR leads to a significant drop in the FID score, especially at higher resolutions. Notably, although affine transformations add only a minimal number of parameters to the model, their role in enhancing image quality is crucial. They deform the 2D coordinate grid at each level, which is essential for accommodating variations in location, size, pose, and deformation.

Table 6 Ablation study on channel dimension of latent vector (z) used during Poly-INR training on ImageNet-100 at different resolutions

z	32^2	64^2	128^2	# Params
768	230	248	272	27.2
896	11.8	18.3	25.7	35.9
1024	5.7	9.6	14.1	45.8
1280	5.2	8.9	13.3	69.2

32^2, 64^2, 128^2 and # Params are same as in Table 5. All Poly-INR models are trained with ten levels

Table 7 An ablation study on the utilization of affine transformation at each level during Poly-INR training on the ImageNet-100 dataset at various resolutions

Affine Transf	32^2	64^2	128^2	# Params
✓	5.7	6.6	8.1	45.82
×	6.5	9.2	14.7	45.81

'Affine Transf.' indicates whether an affine transformation is applied at each level. 32^2, 64^2, 128^2 and # Params are same as in Table 5. All models are trained using ten levels, with the channel dimension of the latent vector (z) set to 1024

4.6 Inference Speed Across Different Resolutions

Table 8 shows the inference speed (sec-per-image) across various resolutions of the Poly-INR model trained on the ImageNet dataset on a Nvidia-RTX-6000 GPU. As discussed in Table 1d, the Poly-INR models use the same number of parameters, i.e., 46 M, at all resolutions. Poly-INR model synthesizes each pixel independently and performs all computations at the same resolution, resulting in slower inference time at higher resolutions.

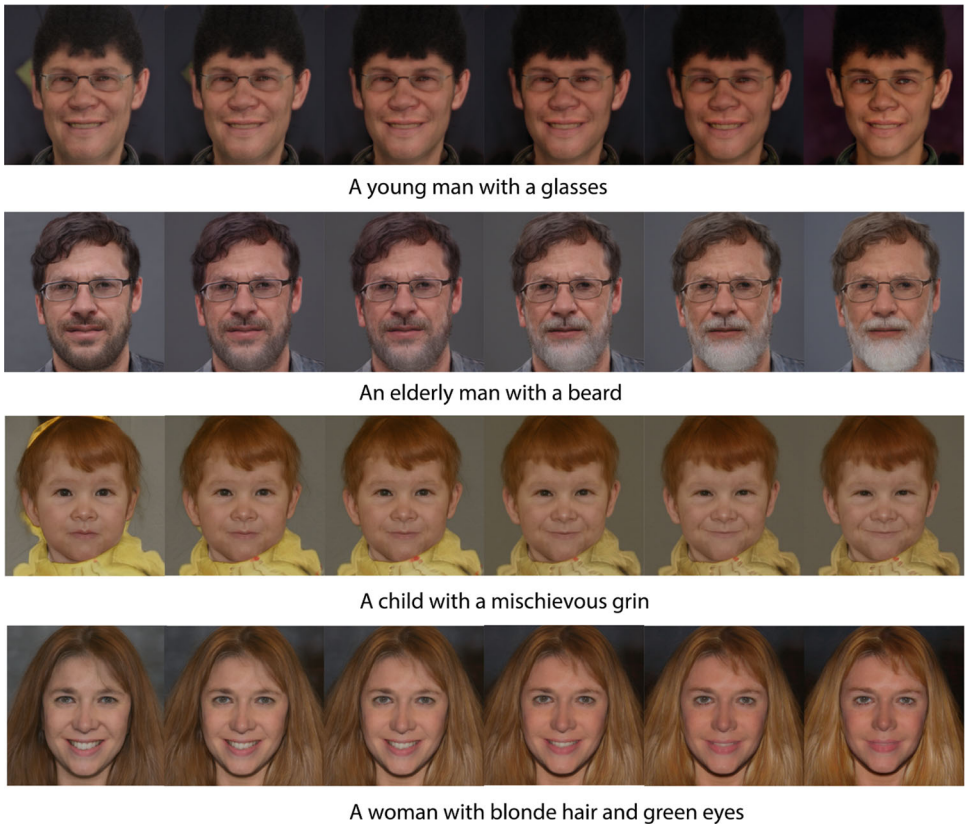

Fig. 10 Illustration of several examples of unique manipulations produced by PolyINR trained on FFHQ dataset at 256 × 256. Each row shows the transitioning of images generated corresponding to the text prompt at several different iterations. We observe gradual transformation for each image

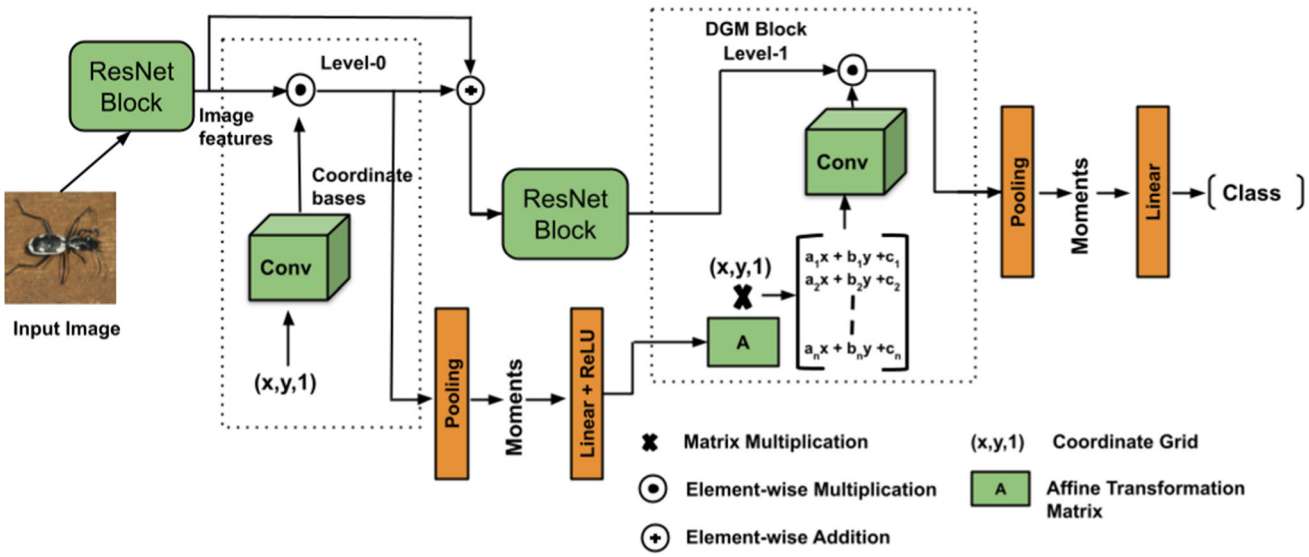

Fig. 11 An overview of the proposed DGM-based framework for the image classification task. The model consists of two blocks, Level-0 and Level-1, that consist of two pipelines: (1) CNN-based image feature extraction and (2) coordinate bases computation. The DGM block can be repeated to increase depth

Table 8 Inference speed (sec-per-image) of Poly-INR model trained on the ImageNet dataset across various resolutions

32^2	64^2	128^2	256^2	512^2
0.007	0.013	0.047	0.179	0.720

5 Further Interpretation and Discussion

In this section, we discuss the significance of the DGM module and evaluate its efficacy in representing intricate geometries and shapes. In Fig. 3, we visualize the heatmap at different levels of our synthesis network. We observe that the initial level forms the basic structure, intermediate levels capture the overall shape, and the higher levels add finer details about the object. To further investigate the prowess of the DGM module, we leverage its capabilities in extracting features for a classification task Singh et al. (2023).

Similar to Poly-INR, our model consists of three components. First, the coordinate base computation which utilizes a 2D coordinate grid as input to generate bases. Second, ResNet blocks to extract image features and lastly, the Affine transformation block to transform the 2D coordinate grid and enable invariance learning.

An overview of the DGM-based model for classification is shown in Fig. 11. The architecture consists of *Level-0* and *Level-1* blocks, where *Level-0* is fixed, whereas *Level-1* can be replicated multiple times to create deeper networks. Level-0 uses the canonical coordinate grid to generate bases and the ResNet block to generate features from the image. We then project this feature on the bases to compute the moments. The projected feature acts as an attention map and is added to the original feature. This feature map and geometric moments are then passed to a ResNet block. The DGM block at Level-1 receives image features from the ResNet block. This level additionally predicts affine parameters based on moments from the preceding level and transforms the coordinate grid to regenerate the bases. The moments from the final level serve as input to a fully connected layer, generating class probabilities for the classification task. Our proposed model does not use any spatial dimension reduction module across the networks. This preserves the shape of the object and thus enhances its interpretability. We also use the same number of feature channels in each ResNet layer for simplicity.

Feature Visualization: To visualize the shape awareness brought by the DGM approach, we visualize the learned features that highlight the object's shape. By the uniqueness theorem Hu (1962), moments can be used to reconstruct the original input, provided the bases are complete. In our case, our learned bases are under-complete. Using the moments as combination weights on the projected features given by:

Table 9 Performance comparison of the DGM model against the standard ResNet model on the ImageNet dataset

Model	Params (M)	Accuracy (%)
ResNet-18	11.69	71.23
DGM ResNet-18	11.88	72.36
ResNet-34	21.80	74.58
DGM ResNet-34	21.32	75.63
ResNet-50	25.56	76.92
DGM ResNet-50	23.51	77.06

$$V = \sum_c m^c (G^c \otimes F^c), \tag{8}$$

Where m^c is the moment, G^c is the basis, F^c is the image feature for channel c, and $\otimes$ is element-wise multiplication, we get a visualization of shape-related information in the features.

Figure 12 compares heatmaps from the DGM-based classification model against the standard ResNet and ViT model on the ImageNet dataset. We use GradCAM Selvaraju et al. (2017) to get visualizations from ResNet-18 models. We also compare our visualization against the attention-based Vision Transformer Dosovitskiy et al. (2020) (ViT-B-16) model. ViT-B is pre-trained on the ImageNet-21K and finetuned on the ImageNet-1K dataset. As shown in Fig. 12, the GradCAM visualizations of the standard ResNet-18 model generate a blob-like shape around the critical region in the image, with no discernible object shape. However, with the DGM-based model, object shapes are crisp, with improved classification accuracies (Table 9). Also, our heat map is much sharper than the vision transformer attention map (Vit-B-16).

Additionally, with the DGM-based model, we can visualize features at different levels providing much better-debugging capability, as shown in Fig. 13. At initial levels, the heatmap is noisy, and the model is not able to able to separate the object from the background. We get much sharper heat maps at higher levels.

Challenges: One of the challenges in our INR method is the higher computation cost compared to the CNN-based generator model for high-resolution image synthesis. The INR method generates each pixel independently; hence, all the computations take place at the same resolution. In contrast, a CNN-based generator uses a multi-scale generation pipeline, making the model computationally efficient. In addition, we observe common GAN artifacts in some generated images. For example, in some cases, it generates multiple heads and limbs, missing limbs, or the object's geometry is not correctly synthesized. We suspect that the

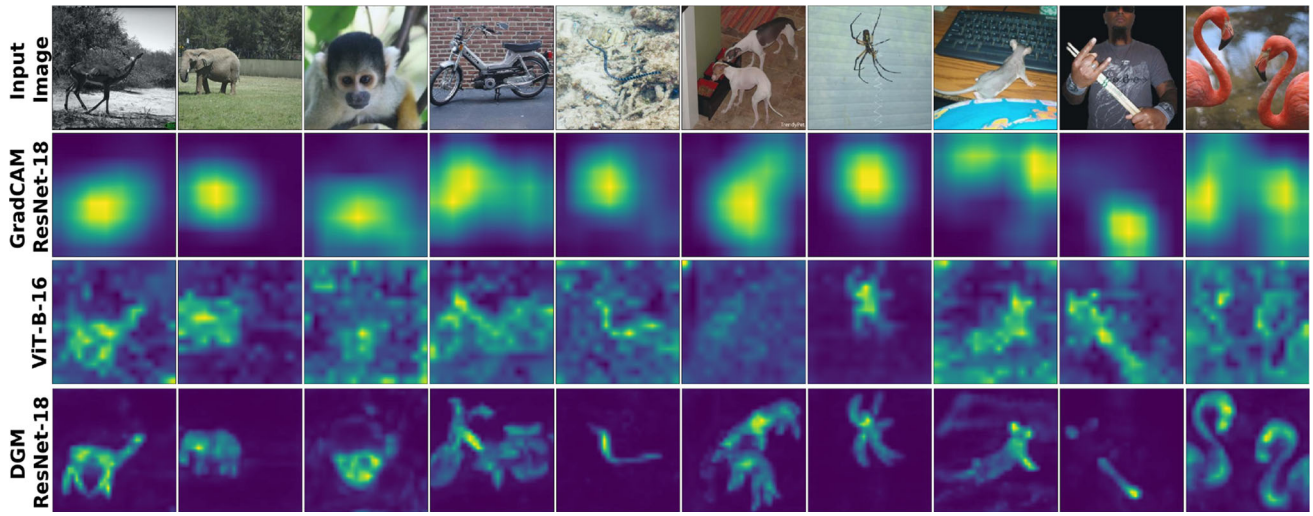

Fig. 12 Feature visualization of different models on ImageNet. For the standard ResNet model, we use GradCAM for visualization. We also compare our visualization with the Vision Transformer Dosovitskiy et al. (2020) (ViT-B-16) attention map

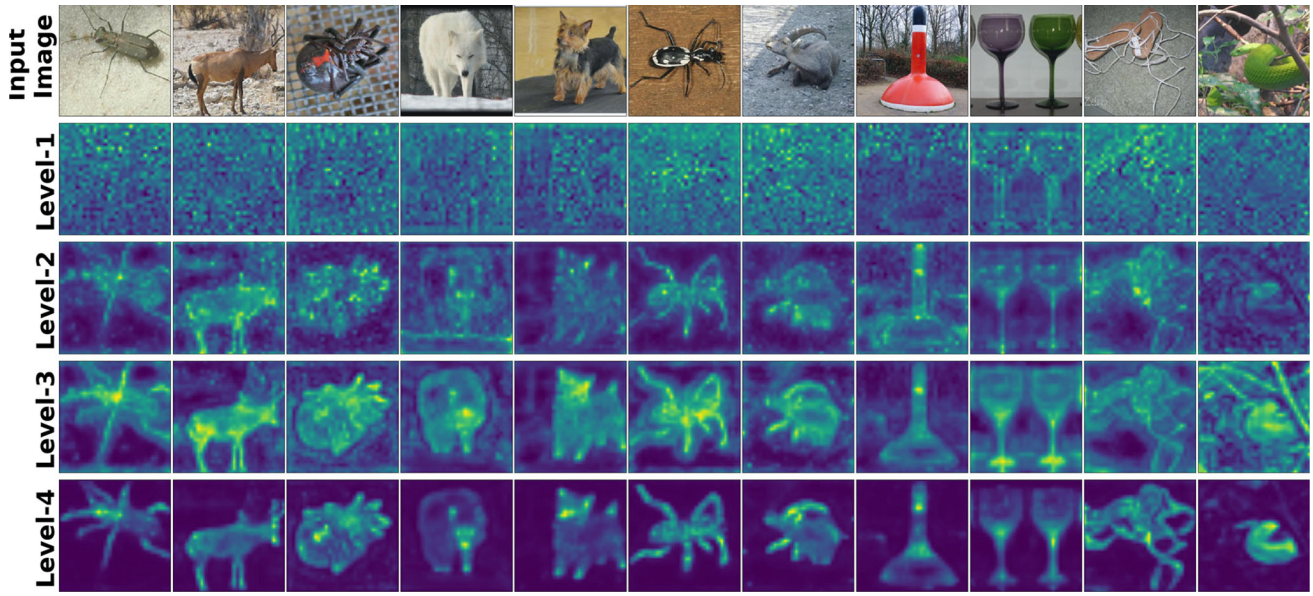

Fig. 13 Visualization at different levels for DGM ResNet-34 model on the ImageNet dataset. We note that at higher levels our model is able to separate the background information from the object's shape compared to initial levels

CNN-based discriminator only discriminates based on the object's parts and fails to incorporate the entire shape.

6 Conclusions and Directions for Future Work

In this work, we propose polynomial function-based implicit neural representations for large image datasets using only linear and ReLU layers. Our Poly-INR model captures high-frequency information and performs comparably to the state-of-the-art CNN-based generative models without using convolution, normalization, upsampling, or self-attention layers. The Poly-INR model outperforms previously proposed positional embedding-based INR GAN models. We demonstrate the effectiveness of the proposed model for various tasks such as interpolation, style-mixing, extrapolation, high-resolution sampling, and image inversion.

Additionally, it would be an exciting avenue for future work to extend our Poly-INR method for 3D-aware image synthesis on large datasets such as ShapeNet. Poly-INR's performance is highly dependent on the latent feature dimension and number of levels. A promising direction for improvement would be to employ neural architecture search to identify the

Table 10 Poly-INR performance on FFHQ-32x32 across various levels (lvl) and model size (number of parameters in Millions)

	Lvl-2	Lvl-4	Lvl-7	Lvl-10	Lvl-14	Lvl-14
Feat. Dim	512	512	512	512	512	1024
Params (M)	2.98	5.62	9.57	13.52	18.79	64.74
FID ↓	27.01	3.46	1.92	1.83	1.52	1.12
Precision ↑	0.85	0.68	0.67	0.68	0.68	0.70
Recall↑	0.01	0.41	0.56	0.57	0.59	0.63

optimal feature dimension and number of levels. In this work, we use the DGM module to promote shape-awareness in the GAN-based model. A simple extension of this work may be integrating geometry-inspired modules with diffusion models.

A Appendix

A.1 Training details

ImageNet: We train the Poly-INR model progressively with increasing resolution. The Poly-INR model is first trained on $200M$ images at 32×32 with 2048 batch size, followed by $72M$ images at 64×64 with 512 batch size, $21M$ images at 128×128 with 256 batch size, $10M$ images at 256×256 with 128 batch size and $2M$ images at 512×512 with 128 batch size. We use learning rate of $1e^{-4}$ for the generator and $2e-4$ for the discriminator. We use Adam optimizer for both the generator and discriminator with $beta = (0.0, .99)$ and $eps = 1e^{-8}$ and the classifier guidance loss weight is set to 8.0 starting at 128×128 and higher resolution. We do not use style mixing regularization and path length regularization.

FFHQ: We also train the Poly-INR model progressively with increasing resolution on the FFHQ dataset. We first train our model with 64×64 on $60M$ images using a batch size of 2048, followed by $15M$ images at 128×128 with 256 batch size and $15M$ images at 256×256 with 256 batch size. The other training hyperparameters are same as the ImageNet experiments described above.

A.2 Ablation Study on the Number of Levels and Feature Dimension

We present an ablation study in Table 10, demonstrating the Poly-INR performance on the FFHQ-32x32 dataset as levels increase. We observe that with increasing levels, the model's performance improves. We utilize 10 levels in our experiments because of training stability and also achieve comparable performance compared to CNN-based models. In case of training with more than 10 levels, we can incrementally increase the number of levels by first training the model on a lower number, such as 10, and gradually add more levels as training progresses.

In Table 10, we increase the model capacity either by adding more levels (layers) or increasing the feature dimension on FFHQ-32x32. We observe that when the model capacity is very small, the recall score is also very poor, but as we increase the model parameters, the recall score gets much better.

A.3 Affine Parameters Mixing

An advantage of representing an image in the polynomial form is that it inherently breaks the image into shape and style. For example, the lower polynomial orders represent the object's shape, whereas the higher orders represent finer details like the style of the image. In our Poly-INR model, manipulating the lower levels' affine parameters changes the object's shape, and manipulating higher levels' affine parameters changes the style. Figure 14 shows examples of style mixing from source A to source B images. In the figure, copying the affine parameters of source A to source B at higher levels (8 and 9) brings fine change in the style, whereas middle levels (5, 6, and 7) bring coarse style change. Figure 15 shows affine parameters mixing at initial levels (0–5). In the figure, we observe that copying the affine parameters at these levels changes the shape of the source B image to the source A image.

A.4 Interpolation

Figure 16 shows linear interpolation between samples of different classes in the affine parameters space. The Poly-INR model provides smooth interpolation between different classes.

A.5 Qualitative Comparison with StyleGAN-XL

We also compare the quality of images generated by Poly-INR model against state-of-the-art CNN-based StyleGAN-XL model for different classes. Figures 17, 18, and 19 show examples of images generated from different classes for the models trained on ImageNet at 256×256. The Poly-INR generates samples qualitatively similar to the StyleGAN-XL model but without using any convolution or self-attention layers.

A.6 Qualitative Comparison with CIPS and INR-GAN on FFHQ Dataset

We also provide qualitative comparison of Poly-INR model against previously proposed INR-based generative models like CIPS and INR-GAN. Figure 20 shows samples gener-

Fig. 14 Source A and B images generated from random latent codes, and the remaining images are generated by copying the affine parameters of source A to source B at different levels. Copying the higher levels' (8 and 9) affine parameters leads to finer style changes, whereas copying the middle levels' (7, 6, and 5) leads to coarse style changes

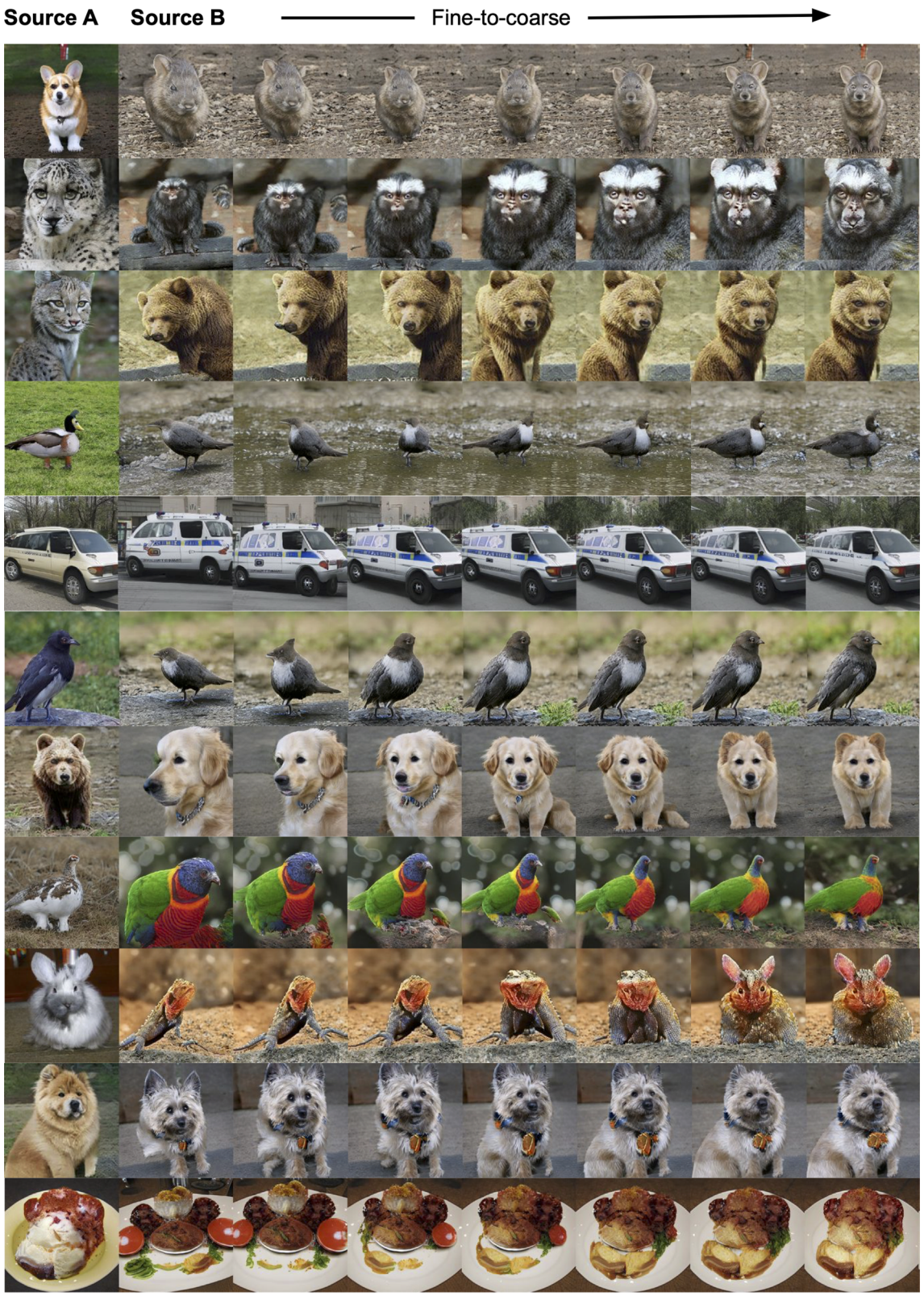

Fig. 15 Source A and B images are generated from random latent codes, and remaining images are generated by copying the affine parameters of source A to source B at different levels. Copying the initial levels' (0, 1, and 2) affine parameters leads to finer shape changes, whereas copying slightly higher levels' (3, 4, and 5) leads to coarse shape changes

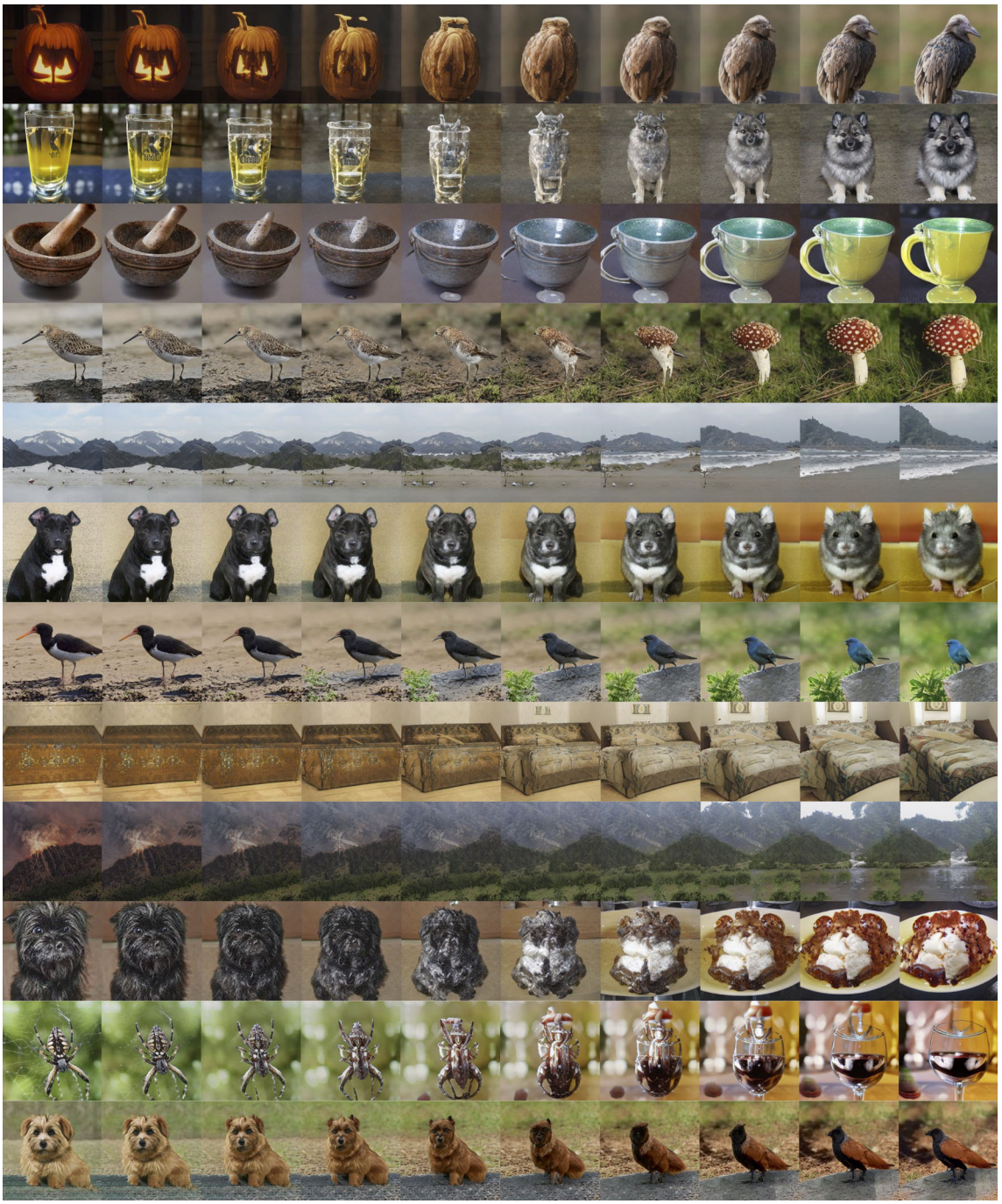

Fig. 16 The poly-INR model generates smooth interpolations between samples of different classes

ated by the three models trained on the FFHQ dataset at 256×256. Our Poly-INR model generates qualitatively better samples than the CIPS and INR-GAN using significantly fewer parameters.

A.7 Qualitative Comparison with Interpolation-Based Upsampling Methods on ImageNet Dataset

In this section, we illustrate the flexibility of Poly-INR to generate images at any desired resolution. We generate higher-resolution images by densely sampling coordinate grids within the $[0, 1]^2$ range. Table 3 presents the FID

Fig. 17 Qualitative comparison between StyleGAN-XL (left column) and Poly-INR (right column). Classes from top to bottom: agaric, daisy, volcano, seashore, cup, and beer glass

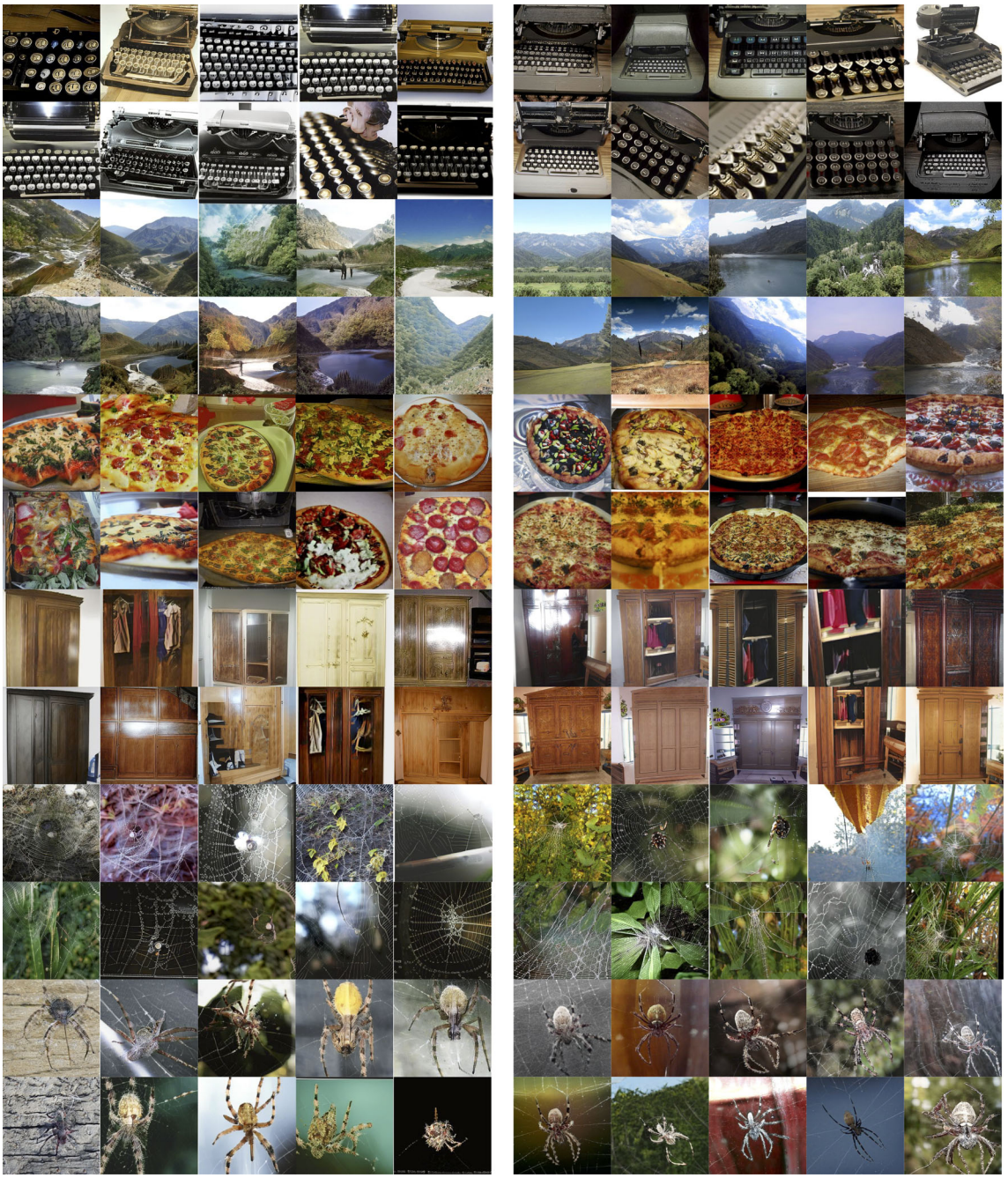

Fig. 18 Qualitative comparison between StyleGAN-XL (left column) and Poly-INR (right column). Classes from top to bottom: type writer, valley, pizza, wardrobe, spider web, barn spider

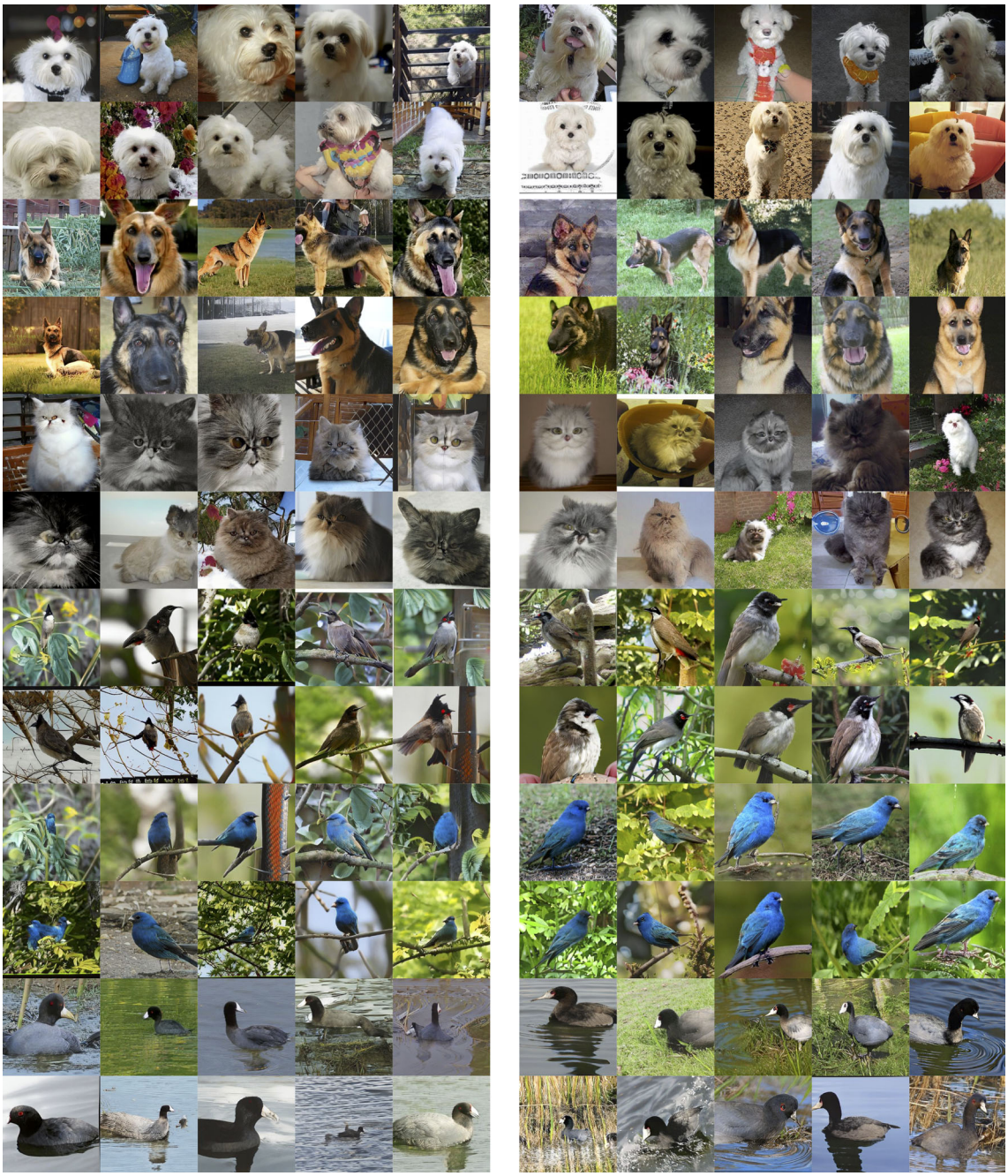

Fig. 19 Qualitative comparison between StyleGAN-XL (left column) and Poly-INR (right column). Classes from top to bottom: maltese dog, german shepherd, persian cat, bulbul, robin, american coot

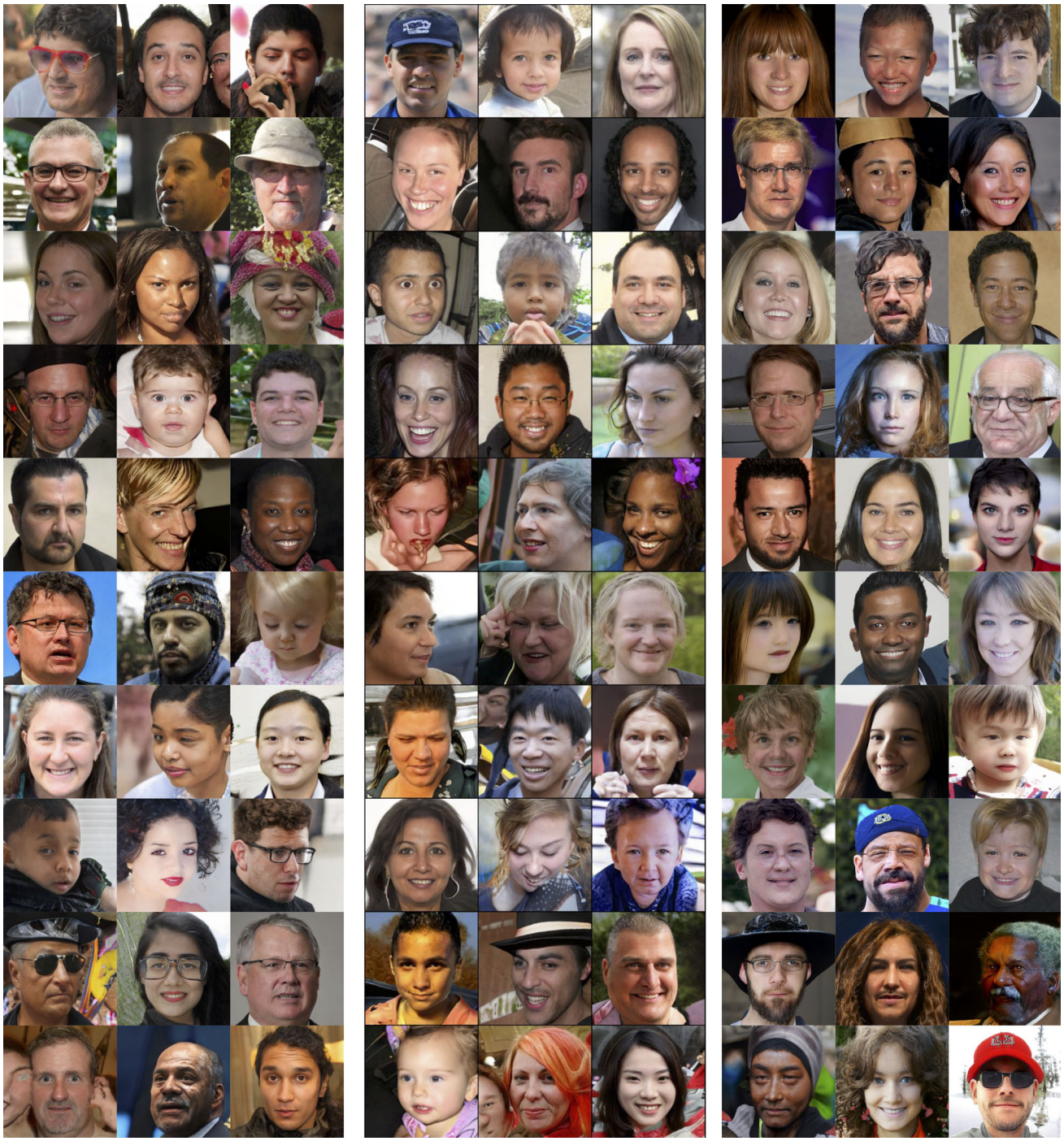

Fig. 20 Qualitative comparison between INR-GAN (left column), CIPS (middle column), and Poly-INR (right column) on FFHQ dataset at 256×256

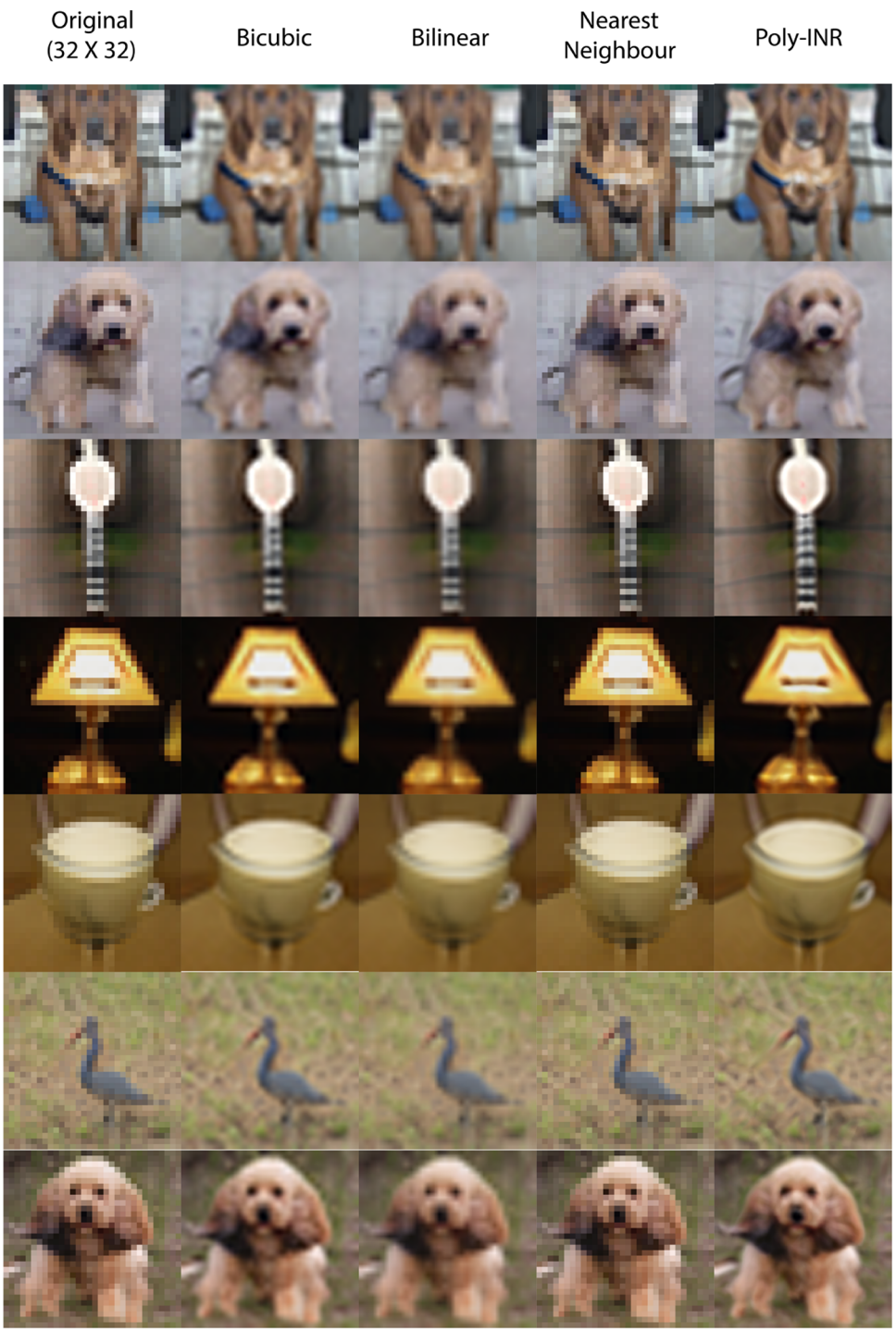

Fig. 21 Qualitative comparison of images upscaled from 32 × 32 to 512 × 512 resolution using classical interpolation-based methods versus Poly-INR, evaluated on the ImageNet dataset. 'Original' denotes images initially generated by Poly-INR at 32 × 32 resolution. Please zoom in for a more detailed examination

Fig. 22 Qualitative comparison of images upscaled from 64 × 64 to 512 × 512 resolution using classical interpolation-based methods versus Poly-INR, evaluated on the ImageNet dataset. 'Original' denotes images initially generated by Poly-INR at 64 × 64 resolution. Please zoom in for a more detailed examination

scores evaluated at 512 × 512 resolution for models initially trained on a lower-resolution ImageNet dataset. In Figs. 21 and 22, we qualitatively compare the quality of images upsampled by our model against those upsampled using classical interpolation-based methods. Among all evaluated methods, Poly-INR and bicubic interpolation produce the highest quality images. Notably, images generated by Poly-INR exhibit superior detail sharpness compared to those upscaled using bicubic interpolation. For instance, consider the dog in the first row of Fig. 21; the left leg of the dog rendered by Poly-INR exhibits significantly greater sharpness compared to its bicubic-interpolated counterpart.

Acknowledgements This work was supported in part by NSF grant 2323086, and by a Subcontract from The Johns Hopkins University with funds provided by an Other Transaction Agreement, No. HR00112490422 from The Defense Advance Research Project Agency. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of The Defense Advance Research Project Agency or The Johns Hopkins University.

References

Alajlan, N., Kamel, M. S., & Freeman, G. H. (2008). Geometry-based image retrieval in binary image databases. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 30*(6), 1003–1013.

Anokhin, I., Demochkin, K., Khakhulin, T., Sterkin, G., Lempitsky, V., & Korzhenkov, D. (2021). Image generators with conditionally-independent pixel synthesis. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 14278–14287.

Arakawa, S., Tsunashima, H., Horita, D., Tanaka, K., & Morishima, S. (2023). Memory efficient diffusion probabilistic models via patch-based generation. arXiv preprint arXiv:2304.07087.

Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein generative adversarial networks. In: *International conference on machine learning (ICML)*, pp. 214–223.

Barron, J., Mildenhall, B., Verbin, D., Srinivasan, P., & Hedman, P. (2022). Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 5470–5479.

Brock, A., Donahue, J., & Simonyan, K. (2018). Large scale GAN training for high fidelity natural image synthesis. In: *International conference on learning representations (ICLR)*.

Chan, E.R., Lin, C.Z., Chan, M.A., Nagano, K., Pan, B., De Mello, S., Gallo, O., Guibas, L.J., Tremblay, J., & Khamis, S., *et al.* (2022). Efficient geometry-aware 3d generative adversarial networks. In: *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition (CVPR)*, pp. 16123–16133.

Chen, Z., & Zhang, H. (2019). Learning implicit fields for generative shape modeling. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 5939–5948.

Chen, Y., Liu, S.,& Wang, X. (2021). Learning continuous image representation with local implicit image function. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 8628–8638.

Chong, C.-W., Raveendran, P., & Mukundan, R. (2004). Translation and scale invariants of Legendre moments. *Pattern Recognition, 37*(1), 119–129.

Chrysos, G. G., Moschoglou, S., Bouritsas, G., Deng, J., Panagakis, Y., & Zafeiriou, S. (2022). Deep polynomial neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 44*(8), 4021–4034.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In: *IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 248–255.

Deng, J., Guo, J., Xue, N., & Zafeiriou, S. (2019). ArcFace: Additive angular margin loss for deep face recognition. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 4690–4699.

Dhariwal, P., & Nichol, A. (2021). Diffusion models beat Gans on image synthesis. *Advances in Neural Information Processing Systems (NeurIPS), 34*, 8780–8794.

Ding, Z., Zhang, M., Wu, J., & Tu, Z. (2023). Patched denoising diffusion models for high-resolution image synthesis. In: *The twelfth international conference on learning representations (ICLR)*.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., & Gelly, S., *et al.* (2020). An image is worth 16x16 words: Transformers for image recognition at scale. In: *International conference on learning representations (ICLR)*.

Du, Y., Collins, K., Tenenbaum, J., & Sitzmann, V. (2021). Learning signal-agnostic manifolds of neural fields. *Advances in Neural Information Processing Systems (NeurIPS), 34*, 8320–8331.

Dupont, E., Kim, H., Eslami, S., Rezende, D., & Rosenbaum, D. (2022a). From data to functa: Your data point is a function and you can treat it like one. In: *International conference on machine learning (ICML)*.

Dupont, E., Teh, Y. W., & Doucet, A. (2022b). Generative models as distributions of functions. In: *International conference on artificial intelligence and statistics (AISTATS)*.

Elad, A., & Kimmel, R. (2003). On bending invariant signatures for surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 25*(10), 1285–1295.

Flusser, J., Boldys, J., & Zitová, B. (2003). Moment forms invariant to rotation and blur in arbitrary number of dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 25*(2), 234–246.

Flusser, J., & Suk, T. (1993). Pattern recognition by affine moment invariants. *Pattern Recognition, 26*(1), 167–174.

Foulonneau, A., Charbonnier, P., & Heitz, F. (2006). Affine-invariant geometric shape priors for region-based active contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 28*(8), 1352–1357.

Gao, S., Zhou, P., Cheng, M.-M., & Yan, S. (2023). Mdtv2: Masked diffusion transformer is a strong image synthesizer. arXiv preprint arXiv:2303.14389.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM, 63*(11), 139–144.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. C. (2017). Improved training of wasserstein gans. *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30.

Härkönen, E., Hertzmann, A., Lehtinen, J., & Paris, S. (2020). Ganspace: Discovering interpretable Gan controls. *Advances in Neural Information Processing Systems (NeurIPS), 33*, 9841–9850.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local Nash equilibrium. Advances in neural information processing systems *30*.

Ho, J., Jain, A., & Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems (NeurIPS), 33*, 6840–6851.

Ho, J., Saharia, C., Chan, W., Fleet, D. J., Norouzi, M., & Salimans, T. (2022). Cascaded diffusion models for high fidelity image generation. *Journal of Machine Learning Research, 23*, 1–47.

Hu, M.-K. (1962). Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory, 8*(2), 179–187.

Joseph-Rivlin, M., Zvirin, A., & Kimmel, R. (2019). Momen(e)t: Flavor the moments in learning to classify shapes. In: *Proceedings of the IEEE/CVF international conference on computer vision workshops*.

Karras, T., Laine, S., & Aila, T. (2019). A style-based generator architecture for generative adversarial networks. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 4401–4410.

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., & Aila, T. (2020). Analyzing and improving the image quality of stylegan. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 8110–8119.

Karras, T., Aittala, M., Laine, S., Härkönen, E., Hellsten, J., Lehtinen, J., & Aila, T. (2021). Alias-free generative adversarial networks. *Advances in Neural Information Processing Systems (NeurIPS), 34*, 852–863.

Khotanzad, A., & Hong, Y. H. (1990). Invariant image recognition by Zernike moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 12*(5), 489–497.

Kim, H. S., & Lee, H.-K. (2003). Invariant image watermark using Zernike moments. *IEEE Transactions on Circuits and Systems for Video Technology, 13*(8), 766–775.

Kynkäänniemi, T., Karras, T., Laine, S., Lehtinen, J., & Aila, T. (2019). Improved precision and recall metric for assessing generative models. In: *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32.

Lee, K., Chang, H., Jiang, L., Zhang, H., Tu, Z., & Liu, C. (2021). Vitgan: Training gans with vision transformers. In: *International conference on learning representations (ICLR)*.

Li, D., Shen, X., Yu, Y., Guan, H., Wang, H., & Li, D. (2020). GGM-net: Graph geometric moments convolution neural network for point cloud shape classification. *IEEE Access, 8*, 124989–124998.

Luciano, L., & Hamza, A. B. (2018). Deep learning with geodesic moments for 3D shape classification. *Pattern Recognition Letters, 105*, 182–190.

Martin-Brualla, R., Radwan, N., Sajjadi, M. S., Barron, J. T., Dosovitskiy, A., & Duckworth, D. (2021). Nerf in the wild: Neural radiance fields for unconstrained photo collections. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 7210–7219.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., & Ng, R. (2021). Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM, 65*(1), 99–106.

Miyato, T., Kataoka, T., Koyama, M., & Yoshida, Y. (2018). Spectral normalization for generative adversarial networks. In: *International conference on learning representations (ICLR)*.

Nash, C., Menick, J., Dieleman, S., & Battaglia, P. (2021). Generating images with sparse representations. In: *International conference on machine learning (ICML)*, pp. 7958–7968.

Ntavelis, E., Shahbazi, M., Kastanis, I., Timofte, R., Danelljan, M., & Van Gool, L. (2022). Arbitrary-scale image synthesis. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11533–11542.

Patashnik, O., Wu, Z., Shechtman, E., Cohen-Or, D., & Lischinski, D. (2021). StyleCLIP: Text-driven manipulation of stylegan imagery. In: *Proceedings of the IEEE/CVF international conference on computer vision (CVPR)*, pp. 2085–2094.

Peebles, W., & Xie, S. (2023). Scalable diffusion models with transformers. In: *IEEE/CVF international conference on computer vision (ICCV)*, pp. 4195–4205.

Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., & Sutskever, I. (2021). Learning transferable visual models from natural language supervision. In: *International conference on machine learning (ICML)*, pp. 8748–8763.

Radford, A., Metz, L. and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv 2015. arXiv preprint arXiv:1511.06434.

Reeves, A. P., Prokop, R. J., Andrews, S. E., & Kuhl, F. P. (1988). Three-dimensional shape analysis using moments and Fourier descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 10*(6), 937–943.

Richardson, E., Alaluf, Y., Patashnik, O., Nitzan, Y., Azar, Y., Shapiro, S., & Cohen-Or, D. (2021). Encoding in style: a StyleGAN encoder for image-to-image translation. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 2287–2296.

Roich, D., Mokady, R., Bermano, A. H., & Cohen-Or, D. (2022). Pivotal tuning for latent-based editing of real images. *ACM Transactions on Graphics (TOG), 42*(1), 1–13.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10684–10695.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., & Berg, A. C. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision, 115*, 211–252.

Sadjadi, F. A., & Hall, E. L. (1980). Three-dimensional moment invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 2*, 127–136.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved techniques for training GANs. In: *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 29.

Sauer, A., Schwarz, K., & Geiger, A. (2022). StyleGAN-XL: Scaling StyleGAN to large diverse datasets. In: *ACM SIGGRAPH*, pp. 1–10.

Sauer, A., Chitta, K., Müller, J., & Geiger, A. (2021). Projected GANs converge faster. *Advances in Neural Information Processing Systems (NeurIPS), 34*, 17480–17492.

Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-cam: Visual explanations from deep networks via gradient-based localization. In: *Proceedings of the IEEE international conference on computer vision (ICCV)*, pp. 618–626.

Singh, R., Shukla, A., & Turaga, P. K. (2023). Improving shape awareness and interpretability in deep networks using geometric moments. In: *Deep Learning in Geometric Computing Workshop (DLGC)*, pp. 4159–4168.

Sitzmann, V., Martel, J., Bergman, A., Lindell, D., & Wetzstein, G. (2020). Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems (NeurIPS), 33*, 7462–7473.

Skorokhodov, I., Ignatyev, S., & Elhoseiny, M. (2021). Adversarial generation of continuous images. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10753–10764.

Skorokhodov, I., Menapace, W., Siarohin, A., & Tulyakov, S. (2024). Hierarchical patch diffusion models for high-resolution video generation. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 7569–7579.

Skorokhodov, I., Sotnikov, G., & Elhoseiny, M. (2021). Aligning latent and image spaces to connect the unconnectable. In: *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, pp. 14144–14153.

Skorokhodov, I., Tulyakov, S., Wang, Y., & Wonka, P. (2022). Epigraf: Rethinking training of 3d GANs. *Advances in Neural Information Processing Systems (NeurIPS), 35*, 24487–24501.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2021). Score-based generative modeling through stochastic differential equations. In: *International conference on learning representations (ICLR)*.

Tan, M., & Le, Q. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. In: *International conference on machine learning (ICML)*, pp. 6105–6114.

Teague, M. R. (1980). Image analysis via the general theory of moments. *Journal of the Optical Society of America (JOSA), 70*(8), 920–930.

Theodoridis, T., Loumponias, K., Vretos, N., & Daras, P. (2021). Zernike pooling: Generalizing average pooling using Zernike moments. *IEEE Access, 9*, 121128–121136.

Tian, Y., Krishnan, D., & Isola, P. (2020). Contrastive multiview coding. In: *European conference on computer vision (ECCV)*, pp. 776–794. Springer

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., & Jégou, H. (2021). Training data-efficient image transformers & distillation through attention. In: *ICML Proceedings of machine learning research*, pp. 10347–10357.

Tuceryan, M. (1994). Moment-based texture segmentation. *Pattern Recognition Letters, 15*(7), 659–668.

Wang, W., Bao, J., Zhou, W., Chen, D., Chen, D., Yuan, L., & Li, H. (2022). Sindiffusion: Learning a diffusion model from a single natural image. arXiv preprint arXiv:2211.12445.

Wang, Z., Jiang, Y., Zheng, H., Wang, P., He, P., Wang, Z., Chen, W., & Zhou, M., *et al.* (2024). Patch diffusion: Faster and more data-efficient training of diffusion models. Advances in Neural Information Processing Systems (NeurIPS) *36*.

Wang, L., & Healey, G. (1998). Using Zernike moments for the illumination and geometry invariant classification of multispectral texture. *IEEE Transactions on Image Processing, 7*(2), 196–203.

Wu, J., Qiu, S., Kong, Y., Chen, Y., Senhadji, L., & Shu, H. (2017). MomentsNet: a simple learning-free method for binary image recognition. In: *IEEE International conference on image processing (ICIP)*, pp. 2667–2671.

Wu, Y., Zhu, Z., Liu, F., Chrysos, G., & Cevher, V. (2022). Extrapolation and spectral bias of neural nets with hadamard product: a polynomial net study. In: *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35.

Yap, P.-T., & Paramesran, R. (2005). An efficient method for the computation of Legendre moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 27*(12), 1996–2002.

Yoon, Y., Chung, I., Wang, L., & Yoon, K.-J. (2022). Spheresr: 360deg image super-resolution with arbitrary projection via continuous spherical image representation. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 5677–5686.

Yu, A., Ye, V., Tancik, M., & Kanazawa, A. (2021). pixelnerf: Neural radiance fields from one or few images. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 4578–4587.

Zhang, H., Shu, H., Coatrieux, G., Zhu, J., Wu, Q. J., Zhang, Y., Zhu, H., & Luo, L. (2011). Affine Legendre moment invariants for image watermarking robust to geometric distortions. *IEEE Transactions on Image Processing, 20*(8), 2189–2199.

Zhang, H., Shu, H., Han, G. N., Coatrieux, G., Luo, L., & Coatrieux, J. L. (2009). Blurred image recognition by Legendre moment invariants. *IEEE Transactions on Image Processing, 19*(3), 596–611.

Zhao, L., Zhang, Z., Chen, T., Metaxas, D., & Zhang, H. (2021). Improved transformer for high-resolution (GANS). *Advances in Neural Information Processing Systems (NeurIPS), 34*, 18367–18380.

Zheng, H., Nie, W., Vahdat, A., Azizzadenesheli, K., & Anandkumar, A. (2023). Fast sampling of diffusion models via operator learning. In: *International conference on machine learning (ICLR)*, pp. 42390–42402. PMLR.

Zheng, H., Wang, Z., Yuan, J., Ning, G., He, P., You, Q., Yang, H., & Zhou, M. (2023). Learning stackable and skippable lego bricks for efficient, reconfigurable, and variable-resolution diffusion modeling. In: *The twelfth international conference on learning representations (ICLR)*.

Zhuang, P., Abnar, S., Gu, J., Schwing, A., Susskind, J. M., & Bautista, M. A. (2022). Diffusion probabilistic fields. In: *International conference on learning representations (ICLR)*.