Adaptive Feature Interpolation for Low-Shot Image Generation

Mengyu Dai[1], Haibin Hang[2], and Xiaoyang Guo[3]

[1] Salesforce
[2] Amazon
[3] Meta
mdai@salesforce.com, haibinh@amazon.com, xiaoyangg@fb.com

Abstract. Training of generative models especially Generative Adversarial Networks can easily diverge in low-data setting. To mitigate this issue, we propose a novel implicit data augmentation approach which facilitates stable training and synthesize high-quality samples without need of label information. Specifically, we view the discriminator as a metric embedding of the real data manifold, which offers proper distances between real data points. We then utilize information in the feature space to develop a fully unsupervised and data-driven augmentation method. Experiments on few-shot generation tasks show the proposed method significantly improve results from strong baselines with hundreds of training samples.

1 Introduction

Majority of learning algorithms today favor the feed of large training data. However, it is often difficult to collect sufficient amount of high-quality data for usage. In addition, intelligent systems like human brains do not need millions of samples to learn useful patterns and are energy-efficient. On the premise of it, learning with small data has been an important research area in various tasks [40, 13, 24, 32, 16, 27, 38, 34, 5]. Among numerous promising works along the direction, a limited amount target on generative models. Training of generative models especially Generative Adversarial Networks (GANs) [11] can easily diverge in low-data setting. To overcome the issue, people come up with methods focusing on different aspects in GAN training, such as data augmentation [17, 43], network architecture design [26, 18] and applying regularization [41, 35]. Data augmentation can substantially increase the size of usable samples and enable stable training [43].

Unlike above data augmentation approaches for generative models which target on image domain, we propose a simple yet effective method to implicitly augment training data without supervision. To our knowledge, it is the first attempt to interpolate the multidimensional output feature of the discriminator for data generation. This can possibly be due to the fact that applications using GAN frameworks usually adopt objectives with 1-dimensional discriminator output, such as vanilla GAN [11] and Wasserstein GAN [2]. Recently, Dai and Hang [8]

introduce a metric learning perspective to the GAN discriminator with multidimensional output and reveal an interesting *flattening effect*: along the training process, the learned metric gradually becomes more uniform and flat. The observation inspire us to explore the possibility of implementing augmentation in feature space in an unsupervised fashion.

In this paper, we propose a way to implicitly augment training data by taking the advantage of the flattening effect of discriminators with multiple output neurons. An intuitive understanding is that, compared to highly sparse and nonlinear nature of real data manifold [3], the low-dimensional feature space is relatively dense and flat. Hence applying interpolation in feature space yields a feasible way for augmentation with higher fidelity. An example of the effect of feature interpolation during training with StyleGAN2 architectures is shown in Figure 1. Both sessions utilized adaptive image augmentation [17], while feature interpolation significantly stabilized training. The novelty of this work is

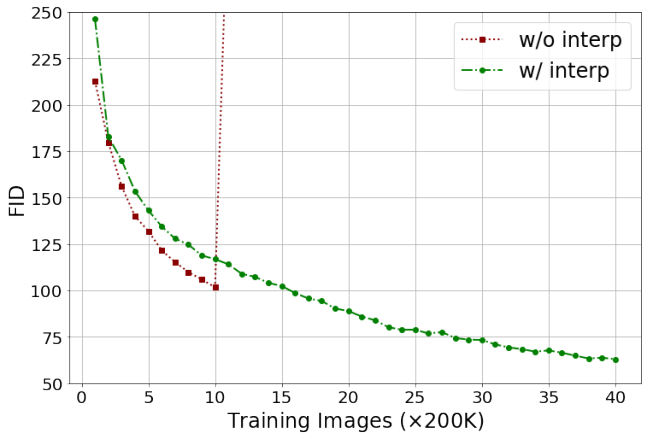


Fig. 1. FID (↓) during training on *Shells* dataset. Red: without feature interpolation; Green: with feature interpolation. Here "training images" refers to training iteration × batch size.

summarized as follows: 1. We propose an implicit data augmentation method in multidimensional feature space of discriminator output. To our knowledge, this is the first attempt in unsupervised image generation. 2. We develop a data-driven approach for augmentation, with criteria based on the underlying structure of feature space during training. 3. Results from few-shot image generation experiments show significant improvements on several benchmark datasets.

2 Related Work

2.1 Low-Shot Data Generation

Recent works contribute to low-shot data generation from different perspectives. Karras *et al.* [17] implemented augmentation on images with adaptive probabil-

ities by using a validation set. Zhao *et al.* [43] proposed DiffAugment which applies random differentiable augmentation on both real and generated images. The above methods significantly improve the amount of available training data in image domain thus remarkably prevent discriminator from overfitting. Tseng *et al.* [35] designed a regularization loss term on predictions of discriminator by tracking the moving averages of discriminator predictions during training. Zhang *et al.* [41] proposed consistency regularization for GANs, with the argument on the invariance of samples after transformation. Liu *et al.* [26] proposed an SLE module and an encoder-decoder reconstruction regularization on discriminator to improve training stability in low-data training settings. For other directions, one can refer to [28, 15, 14, 23]. Different from above techniques, our proposed method is implemented in the multi-dimensional feature space of discriminator output. It does not conflict with data augmentation techniques in the image domain and is independent of network architectures.

2.2 Geometric Interpretations in GANs

The key idea of this paper comes from the interesting flattening effect of discriminator observed in [8]. Particularly, in their paper [8], Dai and Hang interpret the discriminator as a metric generator which learns some intrinsic metric of real data manifold such that the manifold is flat under the learned metric. In this paper, we observe the similar behaviors in the *geometric* GAN [25] framework with hinge loss. Specifically, *geometric* GAN use SVM separating hyper-plane to maximizes the margin between real/fake feature vectors and use hinge loss for discriminator which is simple and fast. Similar effects are also mentioned in [30] by Shao *et al.* which shows manifolds learned by deep generative models are close to zero curvature.

2.3 Interpolation in Feature Space and Mixup

Combining features in the embedding space are shown to be helpful in image retrieval [7, 6, 36, 1]. Recently, Ko *et al.* [21] proposed embedding expansion which utilized a combination of embeddings and performs hard negative mining to learn informative feature representations. DeVries and Taylor [10] claimed that simple transformations to feature space results in plausible synthetic data due to manifold unfolding in feature space. Verma *et al.* [37] introduced Manifold Mixup, which implemented interpolation between hidden feature vectors to obtain smoother decision boundaries at multiple levels of representation. Furthermore, in [37] authors also indicated the flattening effect of Manifold Mixup in learning representations.

Another branch of data augmentation techniques takes advantage of the prior knowledge in learning tasks to interpolate both training images and the corresponding labels. Zhang *et al.* [42] suggested that linear interpolations of data samples should lead to linear interpolations of their associated labels. Kim *et al.* [20] applied batch mixup and formulated the optimal construction of a batch of mixup data. Other works along the track also show improvements on various

discriminative learning tasks [19, 37, 39, 33]. Despite the promising progress, applying these methods require augmentation in training data's associated labels, which does not suit the use case in this paper. Different from above augmentation methods which are mostly used in supervised discriminative tasks, in this work we develop an implicit augmentation method using data-driven feature interpolation, which is suitable for generative tasks in a unsupervised fashion.

3 Methodology

In this section we introduce the main idea of the paper and the proposed implicit augmentation algorithm for low-shot generation in detail.

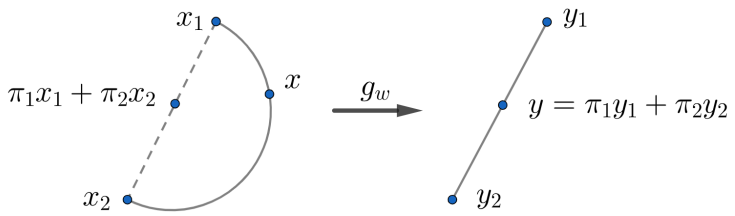


Fig. 2. Direct interpolation of real data likely returns points far away from real data manifold; With flattening effect, we propose interpolation in feature space which returns feature y as an approximation of using some imaginary "real" sample x.

3.1 The Flattening Effect of Discriminator

In this paper, we denote y as a *valid* feature vector iff. $y = g_w(x)$ for some data x from real data manifold given some deep metric learning network g_w. A simple illustration is shown in Figure 2. One question we are interested in is: How far away is the interpolation of a group of valid feature vectors from some individual valid feature vector? In the following we will address this question in a GAN framework using observations from experimental results.

We adopt the default training setting in [26] and *Shells* dataset which contains 64 diverse images for experimentation. [26] uses hinge loss as GAN objective, thus the discriminator has metric learning effect and is equipped with multi-dimensional output. The hinge objective can be formulated as:

$$L_G = \mathbb{E}_{z\sim\mathcal{N}}[-D(G(z))] \tag{1}$$

$$L_D = \mathbb{E}_{x\sim P_{real}}[\max(0, 1 - D(x))] + \mathbb{E}_{x\sim P_{fake}}[\max(0, 1 + D(x))] \ , \tag{2}$$

where $D(x)$ is also named $g_w(x)$ in this paper.

We utilize the following way introduced in [8, 30] to detect the change of the learned metric along the training process: (i) For each iteration i, sample some

real data points to form a finite metric space X_i; (ii) Construct the normalized distance matrix of X_i under the learned metric; (iii) Apply multidimensional scaling (MDS) to the normalized distance matrix to obtain the decreasing (finite) sequence of eigenvalues $C_i = \{\lambda_1^{(i)}, \lambda_2^{(i)}, \cdots, \lambda_b^{(i)}\}$, where b is the number of sample points.

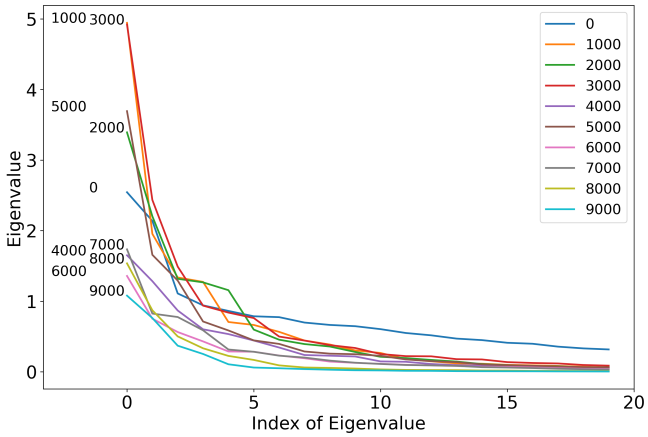


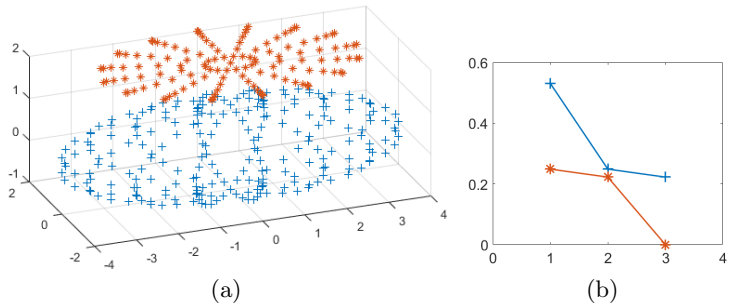
Fig. 3. Each curve represents the first 20 eigenvalues obtained using multidimensional scaling (MDS) of the 64×64 normalized distance matrix from 64 real images under learned metric during training on *Shells* dataset. We draw curves for iterations $0, 1000, \cdots, 9000$.

Now we observe the eigenvalues to see how the Euclidean distance among feature vectors evolve during the training process. As shown in Figure 3, the curve of eigenvalues becomes closer and closer to x-axis with training going forward. At the iteration 9000, only the first few eigenvalues are non-trivial which implies that the valid feature vectors are compressed on a low-dimensional hyperplane. Compared to the input data dimension $m = 1024 \times 1024 \times 3$, the valid feature subspace is significantly flat and uniform. This experimental result is consistent with results in [8], even though the training settings being used are very different. An example of how to infer the shape of data set using eigenvalue curve is shown in Figure 4.

The above observation suggests interesting facts to the question at the beginning of this section: If a set of valid feature vectors $y_1, \cdots, y_k$ are close to each other, then any interpolation point $y = \sum_{i=1}^{k} \pi_i y_i$ with $\sum \pi_i = 1$ and $0 \leq \pi_i \leq 1$ is *very likely* a valid feature vector. Next, we explore how this flattening effect helps with data augmentation.

3.2 Implicit Data Augmentation

Given some neural network g_w, loss function L and training samples x_i, one direct way of augmenting data is to generate synthetic data sample x and use it



(a) (b)

Fig. 4. (a) Blue points are located on an ellipsoid and red points are located on a flat disc; (b) Each curve represents the eigenvalue curve of corresponding point set with the same color.

as training data. All the efforts a synthetic data x could make end up with the calculation of the gradients:

$$\frac{\partial L(g_w(x))}{\partial w} \ .$$

Given valid feature vectors $y_1, y_2, \cdots, y_k$ which are extracted from some training samples $x_1, x_2, \cdots, x_k$. For any interpolation $y = \sum_{i=1}^{k} \pi_i y_i$ with $0 \leq \pi_i \leq 1$ and $\sum_{i=1}^{k} \pi_i = 1$, based on the flattening effect, *very likely* there exists a virtual real data point x such that $y = g_w(x)$. Even though it is not obvious to construct x explicitly, we are able to estimate its contribution to gradients *implicitly* by taking the average of the contributions of $x_1, \cdots, x_k$:

$$\frac{\partial L(g_w(x))}{\partial w} \approx \sum_{i=1}^{k} \pi_i \frac{\partial L(g_w(x_i))}{\partial w} \ ,$$

when $y_1, \cdots, y_k$ are close enough.

The above assertion is summarised in the following:

Lemma 1. *Given some neural network $g_w : \mathbb{R}^m \to \mathbb{R}^n$ and some differentiable loss function $L : \mathbb{R}^n \to \mathbb{R}$. Fix $y = g_w(x)$. Then for a set of nearby points $y_i = g_w(x_i)$, $i = 1, \cdots, k$ such that $y = \sum_{i=1}^{k} \pi_i y_i$, $\sum_{i=1}^{k} \pi_i = 1$, $0 \leq \pi_i \leq 1$, we have:*

$$\left| \frac{\partial L(g_w(x))}{\partial w} - \sum_{i=1}^{k} \pi_i \frac{\partial L(g_w(x_i))}{\partial w} \right| = O(\max_i \|y - y_i\|) \ .$$

Proof. In the following, we use ∂_j to represent the partial derivative of the j-th coordinate.

$$\begin{aligned}
&\frac{\partial L(g_w(x))}{\partial w} - \sum_{i=1}^{k} \pi_i \frac{\partial L(g_w(x_i))}{\partial w} \\
=&\sum_{j=1}^{n} \partial_j L(y) \frac{\partial g_w^{(j)}(x)}{\partial w} - \sum_{i=1}^{k} \pi_i \sum_{j=1}^{n} \partial_j L(y_i) \frac{\partial g_w^{(j)}(x_i)}{\partial w} \\
=&\sum_{i,j} \partial_j L(y) \pi_i \frac{\partial g_w^{(j)}(x_i)}{\partial w} - \sum_{i,j} \pi_i \partial_j L(y_i) \frac{\partial g_w^{(j)}(x_i)}{\partial w} \\
=&\sum_{i,j} \pi_i \frac{\partial g_w^{(j)}(x_i)}{\partial w} \left(\partial_j L(y) - \partial_j L(y_i)\right) \\
=&\frac{\partial g_w(x)}{\partial w} O(\max_i \|y - y_i\|) = O(\max_i \|y - y_i\|)
\end{aligned}$$

In summary, one can update the network parameters by using the average of the gradients raised by some set of training samples when performing gradient descent, if their embedded features are close to each other. In the following sections, we introduce the proposed data-driven augmentation algorithm in detail.

3.3 Nearest Neighbors Interpolation

Denote a data point (image) x_i, its feature vector $y_i = g_w(x_i)$, and the k nearest neighbours (including itself) as $y_{ij}, j = 1, \cdots, k$. We define an interpolated feature for y_i using its k nearest neighbours as

$$\tilde{y_i} = \sum_{j=1}^{k} \pi_{ij} y_{ij} \tag{3}$$

where $\sum_{j=1}^{k} \pi_{ij} = 1$ and $0 \leq \pi_{ij} \leq 1$.

For each y_i, π_{ij} in Eqn (3) follows Dirichlet distribution:

$$\pi_{ij} \sim \mathrm{Dir}(\alpha_{ij}),\ i = 1, 2, \cdots, k \tag{4}$$

One can decide the concentration parameters α_{ij} to control the weights of the nearest neighbors. For example, when $\alpha_{ij} = 1$ for all js, the weights are uniformly distributed. Here we leverage distances between features and geometry of manifold to inform the parameters. The detailed procedure is described as follows.

For the i-th feature y_i and its nearest neighbor $y_{ij},\ j = 1, 2, \cdots, k$:

$$\alpha_{ij} = T(M(y_i, y_{ij}))^t\ , \tag{5}$$

where $T(x) : \mathbb{R}^* \to \mathbb{R}^+$ is a monotonically decreasing function. ($\mathbb{R}^* = \{x \in \mathbb{R}, x >= 0\}, \mathbb{R}^+ = \{x \in \mathbb{R}, x > 0\}$). $M(y_i, y_{ij})$ is the distance between y_i and y_{ij} and $t > 0$ is used to control the skewness of the interpolation. There are lots of choices for $T(x)$, for example, $T(x) = \frac{1}{1+x}$. The intuition is to have larger weights for closer neighbors, as shown in Figure 5. In terms of t, a smaller t gives more uniform/smooth interpolation while larger t prefers more weights on nearer neighbors. For simplicity we set $t = 1$ as the default choice.

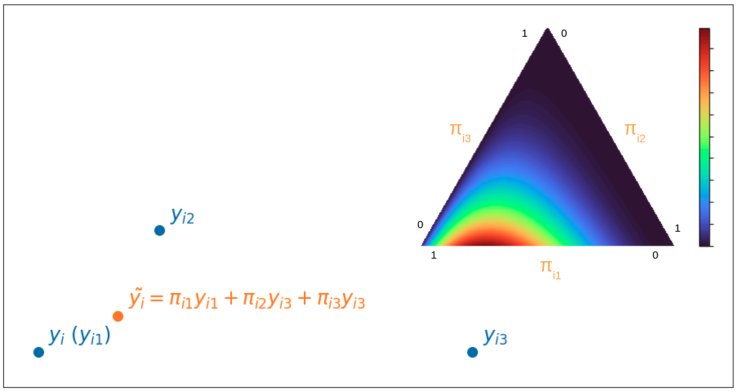


Fig. 5. Illustration of using Dirichlet distribution to interpolate features. $y_{ij}, j = 1, 2, 3$ are the first 3 nearest neighbors of y_i. $\tilde{y_i} = \pi_{i1}y_{i1} + \pi_{i3}y_{i2} + \pi_{i3}y_{i3}$ is the interpolation where π_{ij} are sampled from Dirichlet distribution with concentration parameter α_{ij} in proportion to the distance $M(y_i, y_{ij})$.

3.4 Data-Driven Adaptive Augmentation

To facilitate usage of feature interpolation, we consider taking advantage of the flattening effect during training to decide the aggressiveness of augmentation. Under the context, the aggressiveness can mainly be interpreted by (1) choices of nearest neighbour interpolation, (2) shape of Dirichlet distribution and (3) proportion of augmented features to use. To address (1), when the embedding space is more flat, one may use a larger k for sampling interpolated points. On the contrary, a small k leads to the interpolated features only cover a small portion of the feature space, thus may result in limited augmentation effect and bias in recovering real data manifold. One important question is how to reflect the degree of "flatness" of data manifold. As mentioned earlier, such information can be reached by the (multidimensional scaling) MDS of pairwise distances between features [30, 8]. The flatness can be reflected by the number of large eigenvalues of MDS, where fewer number of large eigenvalues indicate approximately smaller dimensions of the space. Empirically we count number of eigenvalues $\{\lambda_i\}$ bigger than 10% of the largest eigenvalue λ_{max} in a batch b as the effective dimensions, and use $k = I(\lambda_i < 0.1\lambda_{max})$ as number of near neighbours.

The shape of Dirichlet distribution is controlled by M obtained from data itself as discussed in Section 3.3. We also involve augmentation probability p which decides the proportion of interpolated points used for training. $p = 0$ refers to no augmentation, and $p = 1$ refers to when all features are from interpolation. Similarly, we let $p = (k-1)/b$ which introduces more aggressive augmentation with fewer effective dimensions. In practice one can also find other ways to define p, or use fixed ps for simplicity. In experiments we observe using a reasonable choice of p (such as $p = 0.6$) is sufficient for stabilizing training. We will discuss the behaviors of these parameters later in Section 4.2. The whole Adaptive Feature Interpolation (AFI) algorithm is summarized in Algorithm 1.

Algorithm 1 Adaptive Feature Interpolation.

Input: A batch of features $\{y_i\}$ extracted from real data;
Output: Augmented batch of features $\{y_i^*\}$;

1: Calculate distance matrix M from $\{y_i\}$;
2: Solve for MDS of M and return its eigenvalues $\{\lambda_i\}$;
3: Calculate $\{\alpha\}, k, p$ from M and $\{\lambda_i\}$;
4: For each i, sample interpolated features $\tilde{y_i}$ using Eqn (4) with its k near neighbours;
5: For each i, set $y_i^* = \tilde{y_i}$ using Eqn (3) with probability p else $y_i^* = y_i$.

4 Experiments

In this section, we explore the behavior of the proposed method and provide evaluation results on multiple datasets.

4.1 Datasets and Implementation Details

We conducted unconditional generation experiments on several benchmark datasets or their subsets, including Shells, Art, Anime Face, Pokemon provided by [26], Cat, Dog [31], Obama, Grumpy cat [43] and CIFAR-10 [22]. We utilized various metrics for evaluations, including Frechet Inception Distance (FID) [12], Kernel Inception Distance (KID) [4] and Precision and Recall (PR) [29]. By default we generated 50K samples against real data for evaluation. Lower FID, KID scores and higher PR indicate better results.

We adopted StyleGAN2 [17] and FastGAN [26] network architectures, and used consistent parameter settings provided in their papers for experimentation. To facilitate experiments with multidimensional discriminator output, the number of output neurons n of discriminator in [17] was set to 20, and the output logits of discriminator in [26] were reshaped to fit multidimensional setting. Feature interpolation is performed on features extracted from both real and fake images, which is empirically shown to be beneficial in experiments. Experiments were conducted using PyTorch framework on Tesla V100 GPUs.

4.2 Ablation Study

In this section we study the behaviors of proposed feature interpolation (FI) along with experiments using direct image interpolation (II) for comparison.

We first study the effect of p in simple cases using StyleGAN2 architectures [17, 18]. In each setting we recorded FID during training with $p = 0.3, 0.6, 0.9$ as shown in Figure 6. Here in II sessions we performed direct interpolation on images in a *mixup* style [42]. For FI sessions we did not utilize any image augmentation techniques. One can see that compared with FI experiments, the II sessions diverged earlier in all cases. The II session with $p = 0.9$ has less positive effect compared to when $p = 0.6$, which indicates image interpolation does not favor large ps in this case. In contrast, the FI experiments had more stable training sessions even with $p = 0.9$. Best results were obtained with dynamic p in this case. These results suggest that FI may better enjoy the "non-leaking" property [17] in training. In experiments on different datasets we find the use of small amount of original valid features is a necessary regularization for stable training. Note that for II it is not obvious to apply dynamic ks. One reason is that using Euclidean distances between pixels to find near neighbors seems worthless. In addition, with fixed real images one cannot implement dynamic augmentation based on the flattening effect.

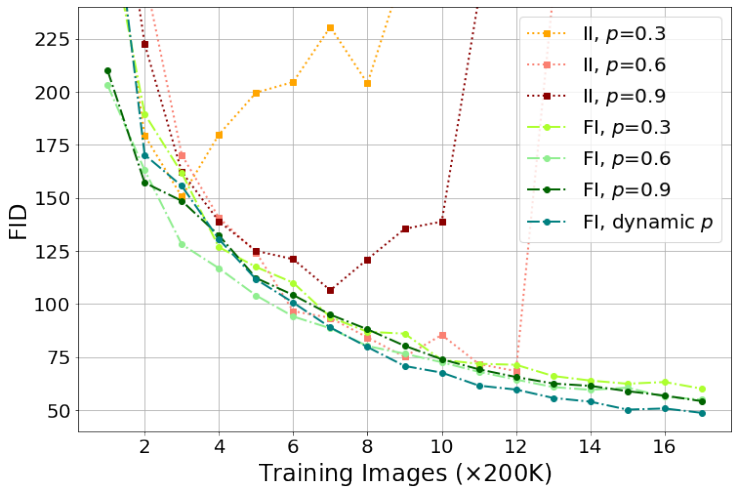


Fig. 6. FID evaluation during training on *Shells* dataset using image interpolation (II) and feature interpolation (FI) with StyleGAN2 architectures.

Next we experimented on settings both uniform ($t = 0$) and skewed ($t = 1$) distributions to study the behavior of Dirichlet distributions on feature interpolation. In each experiment we used batch size 8, and trained 50K iterations with fixed $p = 0.9$ using FastGAN architectures. In this case we employed DiffAug-

ment for image augmentation and used 1K generated images for fast evaluation. In each setting we report the best FID during training along with its corresponding iteration in Table 1. Overall, we notice that FIDs with skewed distri-

	k	1	2	3	4	5	6	7	8
Uniform	FID	165.72	140.89	148.95	141.82	145.58	138.54	136.13	144.90
(t=0)	Iter(K)	10	30	20	40	35	45	35	20
Skewed	FID	165.72	130.98	136.33	141.51	131.54	137.62	139.90	135.85
(t=1)	Iter(K)	10	35	50	40	50	35	35	45

Table 1. FID evaluation of generated samples on *Shells* dataset using FastGAN architectures with fixed ks and $p = 0.9$.

butions are in general better than ones with uniform distributions. Intuitively, with skewed distribution the interpolated features are likely closer to original features, thus may lead to smaller bias in training.

One interesting question is the relation between size of dataset N and value of k in training. We recorded ks up to 500K training images in CIFAR-10 experiments and 200K training images in *Art* experiments with effective batch size 8 on each GPU, and computed averaged k against different Ns. Table 2 also shows

	CIFAR-10				Art	
Images	100	500	5000	50000	100	1000
$p = 0$	4.39	4.02	2.34	2.14	3.41	2.43
AFI	4.62	3.86	2.18	2.26	3.53	2.59

Table 2. Averaged k during training under different amount of training data N with batch size 8 on each GPU.

that under each setting, overall averaged k decreases with N increasing, which corresponds to less augmentation with more real training data. This dynamic mechanism reduces the risk of introducing more biased augmentation with larger N as mentioned in [17], where authors point out an interesting phenomenon that the positive effect of data augmentation decreases with size of real training data increasing. In addition, results in Table 2 also reveal that more training data enlarges the effective dimension of data manifold. Here we also present averaged k during training on CIFAR-10 in Figure 7 which provides some insights on the behaviors of flatness reflected by k. We notice that smaller Ns suggest fewer effective dimensions of data manifold and stronger augmentation. Especially with small Ns ($N = 100$ and 500), the use of augmentation becomes more and more aggressive during training, except for the beginning of training sessions. At the beginning stage, the averaged k fluctuates or even decreases before it reaches the point of inflection. The longer period of this training phase for larger Ns may suggest that the discriminator needs more training to learn meaningful feature representations.

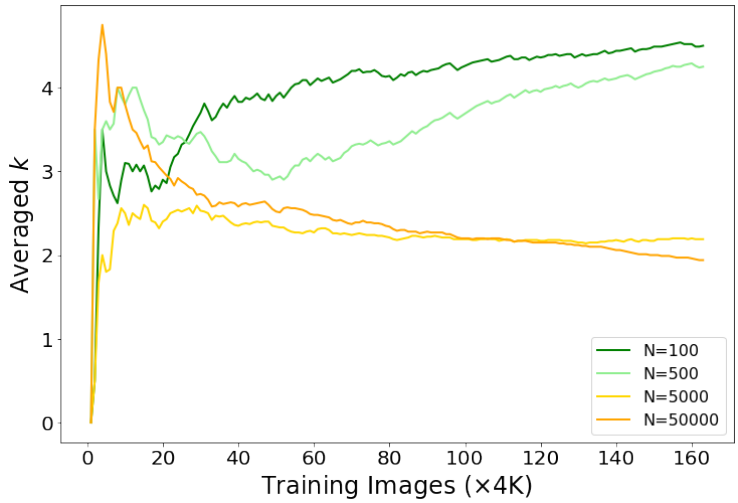


Fig. 7. Averaged k during training on CIFAR-10 with different Ns using effective batch size 8 on each GPU.

To study the effect of feature interpolation on other datasets, we performed experiments using AFI without applying transformation-based image augmentations such as [17, 43]. We also present results using interpolation methods similar as input and manifold mixup [42, 37], except that in the unsupervised task no labels are available for interpolation. In each setting we augmented both real and fake images (or features) with the corresponding method only. Interpolations using [42, 37] were implemented on images and features respectively. Table 3 shows AFI significantly improves baseline results on all datasets. In contrast, directly applying feature interpolation in a *mixup* style led to the worst result.

	Shells	Anime	Art	Pokemon	Cat	Dog	Grumpy cat	Obama
StyleGAN2	229.83	197.01	145.08	194.93	78.32	225.63	43.88	104.64
+ Input	135.66	91.99	86.51	174.91	60.35	118.79	26.98	36.65
+ Feature	319.80	308.16	117.35	376.83	236.99	270.66	118.80	160.57
+ AFI	48.71	85.97	81.08	131.86	57.47	143.60	37.46	31.88

Table 3. FID evaluation of experiments without transformation-based augmentations.

4.3 Results

In the following we provide final evaluations results on various datasets.

FID evaluation of 1024×1024 and 256×256 experiments are displayed in Table 4. In these experiments we incorporated image augmentation techniques,

Dataset	Shells	Anime	Art	Pokemon	Dog	Cat	Grumpy cat	Obama
Image size	1024	1024	1024	1024	256	256	256	256
Number of images	64	120	1000	800	389	160	100	100
FastGAN [26]	152.53	60.04	48.44	57.05	51.24	39.30	27.59	40.52
+ AFI	**124.80**	**55.35**	**43.09**	**50.47**	**50.89**	**35.18**	**25.02**	**36.43**
StyleGAN2 [17]	123.66	60.51	72.36	75.39	59.07	39.78	31.58	44.04
Hinge loss, $n = 20$	101.72	55.67	58.42	55.32	56.43	40.24	28.69	40.18
+ AFI	**62.99**	**33.48**	**43.94**	**44.79**	**49.14**	**35.26**	**22.03**	**31.99**

Table 4. FID evaluations on 1024×1024 and 256×256 experiments using FastGAN and StyleGAN2 architectures.

where [17] applies adaptive augmentation and [26] employs DiffAugment. Table 4 shows that using feature interpolation further improved results from strong baselines. With StyleGAN2 architectures we observe more significant gains across datasets. Note that simply using $n = 20$ without FI, one can already see improvements compared to results from [17]. This behavior is consistent with theoretical analysis and experimental results in [9] that multidimensional discriminator output has its advantage in GAN training. We further present evaluations of KID and precision-recall with StyleGAN2 architectures in Table 5 and Table 6 for reference. Examples of randomly generated 1024×1024 images are displayed in Figure 8. As shown in the figure, samples from experiments with FI have consistently better qualities.

We also tested the effect of feature interpolation on CIFAR-10 with small amount of partial data using the default setting in [17] as baseline. Using feature interpolation improves FID from 42.80 to 27.62 with only 0.2% training data (100 images), and from 19.69 to 13.50 with 1% training data.

	Shells		Anime		Art		Pokemon	
	KID	PR	KID	PR	KID	PR	KID	PR
[17]	20	(0.789,0.085)	15	(0.966,0.933)	26	(0.574,0.823)	28	(0.621,0.727)
+ AFI	**2**	(**0.852**,**0.132**)	**4**	(**0.984**,**0.974**)	**9**	(**0.887**,**0.965**)	**12**	(**0.948**,**0.922**)

Table 5. KID(x10^3)(↓) and Precision-Recall (PR)(↑) evaluations on 1024×1024 experiments.

	Dog		Cat		Grumpy cat		Obama	
	KID	PR	KID	PR	KID	PR	KID	PR
[17]	18	(0.874,**0.948**)	6	(**0.974**,**0.951**)	5	(0.845,0.794)	13	(0.930,0.860)
+ AFI	**14**	(**0.922**,0.932)	**4**	(**0.976**,**0.950**)	**3**	(**0.973**,**0.953**)	**10**	(**0.979**,**0.970**)

Table 6. KID(x10^3) and Precision-Recall (PR) evaluations on 256×256 experiments.

Fig. 8. Randomly generated 1024 × 1024 samples on different datasets. From top to bottom: Shells, Art, Pokemon and Anime. Left: StyleGAN2-ADA [17]; Right: with Adaptive Feature Interpolation.

5 Discussion

In this paper we have proposed an adaptive augmentation approach for low-shot data generation. Instead of producing new training images, the method functions in the multidimensional feature space of discriminator output by utilizing the flattening effect of feature space during training. Experiments show the proposed method improves results from strong baselines in low-data regime.

References

1. Arandjelović, R., Zisserman, A.: Three things everyone should know to improve object retrieval. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. pp. 2911–2918. IEEE (2012)
2. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 70, pp. 214–223. PMLR (06–11 Aug 2017)
3. Balestriero, R., Pesenti, J., LeCun, Y.: Learning in high dimension always amounts to extrapolation (2021)
4. Bińkowski, M., Sutherland, D.J., Arbel, M., Gretton, A.: Demystifying MMD GANs. In: International Conference on Learning Representations (2018)
5. Choe, J., Park, S., Kim, K., Hyun Park, J., Kim, D., Shim, H.: Face generation for low-shot learning using generative adversarial networks. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops (Oct 2017)
6. Chum, O., Mikulik, A., Perdoch, M., Matas, J.: Total recall ii: Query expansion revisited. In: CVPR 2011. pp. 889–896. IEEE (2011)
7. Chum, O., Philbin, J., Sivic, J., Isard, M., Zisserman, A.: Total recall: Automatic query expansion with a generative feature model for object retrieval. In: 2007 IEEE 11th International Conference on Computer Vision. pp. 1–8. IEEE (2007)
8. Dai, M., Hang, H.: Manifold matching via deep metric learning for generative modeling. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 6587–6597 (October 2021)
9. Dai, M., Hang, H., Srivastava, A.: Rethinking multidimensional discriminator output for generative adversarial networks (2021)
10. DeVries, T., Taylor, G.W.: Dataset augmentation in feature space. arXiv preprint arXiv:1702.05538 (2017)
11. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems. vol. 27 (2014)
12. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: Advances in Neural Information Processing Systems. vol. 30 (2017)
13. Hong, J., Fang, P., Li, W., Zhang, T., Simon, C., Harandi, M., Petersson, L.: Reinforced attention for few-shot learning and beyond. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 913–923 (June 2021)
14. Hong, Y., Niu, L., Zhang, J., Liang, J., Zhang, L.: Deltagan: Towards diverse few-shot image generation with sample-specific delta. arXiv preprint arXiv:2009.08753 (2020)
15. Hong, Y., Niu, L., Zhang, J., Zhao, W., Fu, C., Zhang, L.: F2gan: Fusing-and-filling gan for few-shot image generation. In: Proceedings of the 28th ACM International Conference on Multimedia. pp. 2535–2543 (2020)
16. Huang, K., Geng, J., Jiang, W., Deng, X., Xu, Z.: Pseudo-loss confidence metric for semi-supervised few-shot learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 8671–8680 (October 2021)
17. Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., Aila, T.: Training generative adversarial networks with limited data. In: Proc. NeurIPS (2020)

18. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of StyleGAN. In: Proc. CVPR (2020)
19. Kim, J.H., Choo, W., Song, H.O.: Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In: International Conference on Machine Learning (ICML) (2020)
20. Kim, J., Choo, W., Jeong, H., Song, H.O.: Co-mixup: Saliency guided joint mixup with supermodular diversity. In: International Conference on Learning Representations (2021)
21. Ko, B., Gu, G.: Embedding expansion: Augmentation in embedding space for deep metric learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2020)
22. Krizhevsky, A., Nair, V., Hinton, G.: Cifar-10 (canadian institute for advanced research)
23. Li, Y., Zhang, R., Lu, J., Shechtman, E.: Few-shot image generation with elastic weight consolidation. arXiv preprint arXiv:2012.02780 (2020)
24. Li, Y., Zhu, H., Cheng, Y., Wang, W., Teo, C.S., Xiang, C., Vadakkepat, P., Lee, T.H.: Few-shot object detection via classification refinement and distractor retreatment. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 15395–15403 (June 2021)
25. Lim, J.H., Ye, J.C.: Geometric gan (2017)
26. Liu, B., Zhu, Y., Song, K., Elgammal, A.: Towards faster and stabilized {gan} training for high-fidelity few-shot image synthesis. In: International Conference on Learning Representations (2021)
27. Liu, S., Wang, Y.: Few-shot learning with online self-distillation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops. pp. 1067–1070 (October 2021)
28. Ojha, U., Li, Y., Lu, C., Efros, A.A., Lee, Y.J., Shechtman, E., Zhang, R.: Few-shot image generation via cross-domain correspondence. In: CVPR (2021)
29. Sajjadi, M.S.M., Bachem, O., Lučić, M., Bousquet, O., Gelly, S.: Assessing Generative Models via Precision and Recall. In: Advances in Neural Information Processing Systems (NeurIPS) (2018)
30. Shao, H., Kumar, A., Thomas Fletcher, P.: The riemannian geometry of deep generative models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (June 2018)
31. Si, Z., Zhu, S.C.: Learning hybrid image templates (hit) by information projection. IEEE Transactions on Pattern Analysis and Machine Intelligence **34**, 1354–1367 (2012)
32. Stojanov, S., Thai, A., Rehg, J.M.: Using shape to categorize: Low-shot learning with an explicit shape bias. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1798–1808 (June 2021)
33. Thulasidasan, S., Chennupati, G., Bilmes, J.A., Bhattacharya, T., Michalak, S.: On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In: Advances in Neural Information Processing Systems. vol. 32. Curran Associates, Inc. (2019)
34. Tian, Y., Wang, Y., Krishnan, D., Tenenbaum, J.B., Isola, P.: Rethinking few-shot image classification: a good embedding is all you need? ArXiv **abs/2003.11539** (2020)
35. Tseng, H.Y., Jiang, L., Liu, C., Yang, M.H., Yang, W.: Regularing generative adversarial networks under limited data. In: CVPR (2021)

36. Turcot, P., Lowe, D.G.: Better matching with fewer features: The selection of useful features in large database recognition problems. In: 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops. pp. 2109–2116. IEEE (2009)
37. Verma, V., Lamb, A., Beckham, C., Najafi, A., Mitliagkas, I., Lopez-Paz, D., Bengio, Y.: Manifold mixup: Better representations by interpolating hidden states. In: International Conference on Machine Learning. pp. 6438–6447. PMLR (2019)
38. Yao, H., Wu, X., Tao, Z., Li, Y., Ding, B., Li, R., Li, Z.: Automated relational meta-learning. In: International Conference on Learning Representations (2020)
39. Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y.: Cutmix: Regularization strategy to train strong classifiers with localizable features. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (October 2019)
40. Zhang, C., Song, N., Lin, G., Zheng, Y., Pan, P., Xu, Y.: Few-shot incremental learning with continually evolved classifiers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 12455–12464 (June 2021)
41. Zhang, H., Zhang, Z., Odena, A., Lee, H.: Consistency regularization for generative adversarial networks. In: International Conference on Learning Representations (2020)
42. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. International Conference on Learning Representations (2018), https://openreview.net/forum?id=r1Ddp1-Rb
43. Zhao, S., Liu, Z., Lin, J., Zhu, J.Y., Han, S.: Differentiable augmentation for data-efficient gan training. In: Conference on Neural Information Processing Systems (NeurIPS) (2020)