Implicit Data Augmentation Using Feature Interpolation for Diversified Low-Shot Image Generation

Mengyu Dai
Microsoft
Haibin Hang
University of Delaware
Xiaoyang Guo
Meta
{dz1d00511,haibin.hang312,xiaoyang.guo.1}@gmail.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 diverse samples. 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 data-driven augmentation method. We further bring up a simple metric to evaluate the diversity of synthesized samples. Experiments on few-shot generation tasks show our method improves FID and diversity of results compared to current methods, and allows generating high-quality and diverse images with less than 100 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 [7, 16, 19, 28, 31, 35, 37, 41, 44]. Among numerous promising works along the direction, a limited amount target on generative models. Training of generative models especially Generative Adversarial Networks (GANs) [13] 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 [20, 48], network architecture design [22, 30] and applying regularization [38, 46]. Data augmentation can substantially increase the size of usable samples and enable stable training [48].

Unlike the above data augmentation approaches for generative models which target on image domain, in this paper we propose a simple yet effective method to augment data implicitly by interpolating the multi-dimensional output feature of the discriminator. To our knowledge, this is the first attempt to apply implicit data augmentation in feature space for low-shot 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 [13] and Wasserstein GAN [2]. Recently, Liu et al. [30] propose a strong network architecture equipped with hinge loss using multi-dimensional discriminator output. Their proposed SLE modules make it possible to generate high-resolution and visually appealing samples using only hundreds of samples. Dai and Hang [11] introduce a metric learning perspective to the GAN discriminator with multi-dimensional output and reveal an interesting flattening effect: along the training process, the learned metric gradually becomes more uniform and flat. These works inspire us to explore the possibility of implementing augmentation in feature space for few-shot data generation.

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 non-linear 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. A simple illustration is shown in Figure 1. Another novelty in this work is that we propose a simple metric to quantify diversity of gener-
ated samples, as a supplement to existing evaluation methods such as Fréchet Inception Distance (FID) and Inception Score (IS). Chong et al. [8] shows FID and IS are biased for a finite sample set, and the bias term also depends on the particular model being evaluated. Under low-data setting, the amount of trustworthy information one can obtain from these metrics is even more limited.

We summarize the novelty of this work as follows: 1. We propose an implicit data augmentation method in multi-dimensional feature space of discriminator output. To our knowledge, this is the first attempt in low-shot data generation with GANs. 2. We develop a data-driven approach for dynamic augmentation. The augmentation criteria is based on the underlying structure of feature space during training. 3. We bring up a simple diversity metric for image generation, which serves as an supplement to common metrics to help evaluating diversity of synthesized samples. 4. We show performance improvement on several benchmark datasets for few-shot generation. The proposed method can generate high-quality samples with fewer training images.

2. Related Work

2.1. Low-Shot Data Generation

Recent works contribute to low-shot data generation from different perspectives. Karras et al. [20] implemented augmentation on images with adaptive probabilities by using a validation set. Zhao et al. [48] 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. [38] designed a regularization loss term on predictions of discriminator by tracking the moving averages of discriminator predictions during training. Zhang et al. [46] proposed consistency regularization for GANs, with the argument on the invariance of samples after transformation. Liu et al. [30] 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 [17, 18, 27]. 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 [11]. Particularly, in their paper [11], 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 [29] framework with hinge loss. Specifically, geometric GAN use SVM separating hyper-plane to maximize the margin between real/fake feature vectors and use hinge loss for discriminator which is simple and fast. Similar effects are also mentioned in [33] 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 [1, 9, 10, 39]. Recently, Ko et al. [25] proposed embedding expansion which utilized a combination of embeddings and performs hard negative mining to learn informative feature representations. DeVries and Taylor [12] claimed that simple transformations to feature space results in plausible synthetic data due to manifold unfolding in feature space. Verma et al. [40] introduced Manifold Mixup, which implemented interpolation between hidden feature vectors to obtain smoother decision boundaries at multiple levels of representation. Furthermore, in [40] 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. [45] suggested that linear interpolations of data samples should lead to linear interpolations of their associated labels. Kim et al. [23] 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 [24, 36, 40, 43]. 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. Moreover, directly applying interpolation on images could largely twist the real data distribution, thus is not ideal for the goal of a generative task aiming at learning distribution from data itself [48]. 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 an 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.

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 $g_w$. One question we
are interested in: 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 [30] and Shells dataset which contains 64 diverse images for experimentation. [30] 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:

\[
L_G = \mathbb{E}_{z \sim \mathcal{N}}[-D(G(z))] \\
L_D = \mathbb{E}_{x \sim P_{real}}[\max(0, 1 - D(x))] + \mathbb{E}_{x \sim P_{fake}}[\max(0, 1 + D(x))] \tag{1}
\]

We utilize the following way introduced in [11,33] 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 \(d_i\); (iii) Apply multidimensional scaling to \(M_i\) to obtain the decreasing (finite) sequence of eigenvalues \(C_i\).

Now we observe the eigenvalues to see how the Euclidean distance among feature vectors evolve during the training process. As showed in Figure 2, 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 \(m = 1024 \times 1024 \times 3\) and feature space dimension \(n = 50\), the valid feature subspace is significantly flat and uniform. This experimental result is consistent with results in [11], 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 3.

The above observation suggests interesting facts to the question at the beginning of this section: If a set of 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 contributes to gradients implicitly by taking average of the contributes 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 3.1.** 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 ≤ π_i ≤ 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 \( \partial_j \) to represent the partial derivative of the \( j \)-th coordinate.

\[
\begin{align*}
\frac{\partial L(g_w(x))}{\partial w} &= \sum_{i=1}^{k} \pi_i \frac{\partial L(g_w(x_i))}{\partial w} \\
&= \sum_{j=1}^{n} \frac{\partial L(y)}{\partial w} \frac{\partial g_w(y_j)}{\partial w} - \sum_{i=1}^{n} \pi_i \frac{\partial L(y_i)}{\partial w} \frac{\partial g_w(y_i)}{\partial w} \\
&= \sum_{i,j} \pi_i \frac{\partial g_w(y_i)}{\partial w} (\partial_j L(y) - \partial_j L(y_i)) \\
&= \frac{\partial g_w(x)}{\partial w} O(\max_i \|y - y_i\|) = O(\max_i \|y - y_i\|)
\end{align*}
\]

\[ \Box \]

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

\[
\tilde{y}_i = \sum_{j=1}^{k} \pi_{ij} y_{ij}
\]

where \( \sum_{j=1}^{k} \pi_{ij} = 1 \) and \( 0 ≤ \pi_{ij} ≤ 1 \).

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

\[
\pi_{ij} \sim \text{Dir}(\alpha_{ij}), \ i = 1, 2, \cdots, k
\]

One can tune the concentration parameters \( \alpha_{ij} \) to control the weights of the nearest neighbors. For example, when \( \alpha_{ij} = 1 \) for all \( j \), the weights are uniformly distributed. We can also leverage distances between embeddings and geometry of manifold to inform the parameters. The detailed procedure as 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 \( T(x) : \mathbb{R}^+ \rightarrow \mathbb{R}^+ \) is a monotonically decreasing function. (\( \mathbb{R}^+ = \{x \in \mathbb{R}, x > 0\}, \mathbb{R}^+ = \{x \in \mathbb{R}, x \geq 0\} \)). \( M(y_i, y_{ij}) \) 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 4. In terms of \( t \), a smaller \( t \) gives more uniform/smooth interpolation while larger \( t \) prefers more weights on nearer neighbors. We will further discuss the choice of these parameters in the next section.

### 3.4. Data-Driven Dynamic Augmentation

We now 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. Although one can fix \( p \) for simple use, we consider involving a dynamic mechanism to select \( p \) which takes the “flattening” information of data manifold during training into account. As mentioned earlier, a flat local structure around some \( y \) leads to small bias of using linear combination of its neighbours. Thus when the data manifold is more flat, we tend to assign higher probabilities to augmented features. The key question becomes how to reflect the degree of “flatness” of data manifold. As mentioned earlier, such information can be reached by the MDS of pairwise distances between features [11, 33]. Smaller and more uniform eigenvalues of MDS indicate the shape of manifold is more “flat”, where we tend to assign a higher \( p \), otherwise we tend to use a smaller \( p \) to reduce the risk of introducing more bias from interpolation. Denote \( M \) as pairwise distance matrix of \( \{y_i\} \), and the eigenvalues of its MDS as \( \{\sigma_i\} \). We define
dynamic probability augmentation as:

\[ p = \left( \frac{1}{e^{\sigma_{\text{max}}}} \right)^{\frac{k}{r}} \]  

(6)

where \( r \) is a user-defined parameter to control the aggressiveness of applying augmentation. A smaller \( r \) leads to more aggressive augmentation. \( \sigma_{\text{max}} \) denotes the largest eigenvalue and \( \sigma \) denotes the mean of eigenvalues.

For the use of Dirichlet distribution in feature interpolation, one can control the skewness of the distribution by adjusting the penalty \( t \) with training going forward. One simple way to decide a dynamic \( t \) is:

\[ t = (1 - p)^s \]  

(7)

where \( s \in \mathbb{R} \). When \( s \to -\infty \), we do not apply any augmentation on original data; When \( s \to +\infty \), the Dirichlet distribution becomes uniform; When \( s = 0 \) we do not apply penalty on \( \alpha \). In practice, one can design other ways to formulate dynamic parameters based on different criteria.

As to the choice of \( k \), a small \( k \) leads to the interpolated features only cover a small portion of the feature space, thus may result in limited augmentation ability and bias in recovering real data manifold. With proper use of Eqn (5) and (7), one can use a large \( k \) meanwhile control the distribution of interpolated features. In the low-data setting, one simple way is to let \( k \) equal to the actual training batch size. The whole implicit dynamic augmentation algorithm is summarized in Algorithm 1.

Algorithm 1 Implicit Dynamic Augmentation.

Input: \( k, r, s \) and a batch of features \( \{ y_i \} \);  
Output: Augmented batch of features \( \{ \tilde{y}_i \} \);

1: Calculate distance matrix \( M \) from \( \{ y_i \} \);  
2: Solve for MDS of \( M \) and return its eigenvalues \( \{ \sigma_i \} \);  
3: Calculate \( \{ \alpha \}, p, t \) from \( M, r, s \) and \( \{ \sigma_i \} \) using Eqn (5,6,7);  
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 \( y_i^* = y_i \).

3.5. Diversity Measurement of Synthesized Samples

Common metrics for generative models such as Inception Score (IS) [32] and Fréchet Inception Distance (FID) [14] are shown to have non-negligible bias with limited samples or may not detect overfitting [4,5,8]. Such behavior of FID and IS can lead to less informative evaluation in low-data settings. In low-data cases models can suffer the risk of achieving appealing FID scores while generated sampled may not well cover all types of real samples. This mode dropping behavior can be tricky in evaluation of generative models. Inspired by discussions in [30] which utilizes LPIPS [47] between individual synthesized image and its nearest real sample, we quantify the overall diversity of \( N_F \) generated data points \( \tilde{x}_i \) based on \( N_R \) referenced real data points \( x_j \) as follows:

\[ \text{DScore} = \frac{1}{N_R} \sum_{j=1}^{N_R} \sum_{i=1}^{N_F} \text{LPIPS}(\tilde{x}_i, x_j) \]  

(8)

where \( l_j \) represents the number of generated samples with \( x_j \) as the nearest real sample. Higher DScore indicates generated samples are more diverse.

4. Experiments

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

4.1. Datasets and Evaluation Metrics

We apply our method on several benchmark datasets (or their subsets) for experimentation, including Cat, Dog [34], Obama, Panda, Grumpy cat [48], Shells, Art, Fauvism, Anime Face provided by [30], LSUN church, LSUN bridge [42], FFHQ [21], CIFAR-10 and CIFAR-100 [26]. We use Fréchet Inception Distance (FID) [15] to evaluate the quality of synthesized samples. We also include DScore as an additional measurement of diversity of generated images as introduced in Section 3. For both metrics we use 1000 synthesized data against real data set for evaluation. Lower FID and higher DScore indicate better results.

4.2. Implementation Details

For experiments with \( 1024 \times 1024 \) and \( 256 \times 256 \) images we use the default training settings in [30] which is currently state-of-the-art method on low-shot image generation. For CIFAR-10 and CIFAR-100 experiments we use BigGAN [6] architecture and the default hyper-parameters. Detailed parameter settings can be found in supplementary material. We use DiffAugment as a direct augmentation method in image domain to obtain strong baselines. We then apply the proposed implicit augmentation on features extracted from both real and fake images, which is shown to be beneficial in experiments. Experiments were conducted using PyTorch framework on single Tesla V100 GPU.

4.3. Ablation Study on \( k \) and \( p \)

We use Shells dataset as our default choice for ablation study. In practice one can also follow the hints to try out on user’s own datasets. We first study the effect of \( p \) in simple cases where \( k = 2 \) against baseline setting without feature interpolation (the same as in [30]). For each setting
we train 50K iterations and present FID during training with $p = 0, 0.3, 0.6, 0.9$ in Figure 5(a). One can see the baseline setting soon reaches its best FID at 10K iterations and diverges quickly afterwards. In contrast, the other sessions with feature interpolation manage to go forward stably for a longer time. The session with $p = 0.9$ is more stable compared to other ones, which indicates the benefit of using a relatively large proportion of interpolated features. We have also tried using $p = 1$, but did not find satisfying and stable sessions. Indeed, in experiments we find the use of small amount of original valid features is a necessary regularization for stable training. In practice, one simple criteria to choose $p$, if not use dynamic ones, can be decided by user’s computational budget. For example, use a small $p$ with limited training time, or a relatively large $p$ to potentially find better local minimum in a longer training session.

Next we study the behavior of $k$ under two types of distributions for interpolated features: uniform and skewed. We set $t = 0$ to obtain uniform distributions, and $t = 1$ to obtain skewed distributions, respectively. For each type of distribution, we use batch size 8 for cases when $k \leq 8$, batch size 16 for $k = 16$, and train 50K iterations under different $k$s with fixed $p = 0.9$. In each setting we report the best FID during training along with its corresponding iteration and DScore in Table 1. We observe in cases such as when $k = 4$ with uniform distribution, although the FID (141.82) is lower compared to baseline score (165.86), its DScore (0.0397) is almost the worst among all training settings. Thus for this session the mode dropping issue was hidden from FID but detected by DScore. Overall, we notice that the best FIDs with skewed distributions ($t = 1$) are in general better than ones with uniform distributions ($t = 0$); On the contrary, the top DScores are from settings with uniform distributions ($k = 5, 8, 16$). Intuitively, with skewed distribution, the interpolated features are likely closer to original features, thus with small bias but introduces limited diversity; With uniform distribution, the chance of selecting interpolated features far away from original features is higher, and in this way it introduces more diversity meanwhile larger bias. Hence it would be beneficial if the shape of the distribution can be controlled in a dynamic data-driven fashion.

### 4.4. Study on Dynamic Augmentation

As presented in Figure 2, the eigenvalues decrease in scale and become more uniform during training. This behavior allows us to assign augmentation more aggressively in the more “flat” space. Here we use Eqn (6) to calculate dynamic augmentation probability $p$ under different $r$s as shown in Figure 5(b). As one can see, with training going forward $p$ gradually increases in all cases. Next we study the behavior of $s$ which controls the distribution of inter-
polated features. We fix $r = 0.2$ to result in large $p$s and train 150K iterations under different values of $s$. Figure 5(c) shows recorded FIDs every 5K iterations. With large values of $p$, $t = -1$ introduces highly skewed distribution of interpolated features, thus leads to little augmentation effect. With $s$ getting larger, distributions of interpolated features become more uniform. We present the evaluation results along with DScores and averaged augmentation probability $\overline{p}$ in Table 3. For $\overline{p}$ we report the average over every 100 training iterations. The standard deviations for the four cases are 0.0314, 0.0330, 0.0357 and 0.0350 respectively.

Here we also plot the distribution of fake images to nearest real images in Figure 6 using above experimental results. One can see with higher DScore (Figure 6(d)) the corresponding distribution becomes more uniform, where generated samples are separately close to different real images. The distribution centered to fewer real samples leads to a lower score (Figure 6(b)).

### 4.5. Results

We present evaluation results of $1024 \times 1024$ images on several benchmark datasets in Table 2. Under the same settings the proposed method improved baseline evaluation results on most of the datasets. We see more significant improvements on datasets with fewer training samples or diverse distributions, such as Art, Cat and Shells. We observe mild improvements on datasets with simple distributions, such as Obama, Panda and Grumpy cat, where baseline results are already very impressive. Examples of generated $1024 \times 1024$ samples are displayed in Figure 7. One can see that randomly generated samples using the proposed method can cover more subcategories on different datasets.

We then conducted experiments on $256 \times 256$ images with more training samples. Table 4 shows the proposed method improved baseline results on the three datasets. For fair comparison we used the same training setting and trained up to 150K iterations on each dataset for both image sizes. In practice, with more training data one can conduct longer training sessions to potentially obtain better local minimum.

We also tested the effect of the proposed method on CIFAR-10 and CIFAR-100 with partial data using BigGAN architecture. Results from using 5K, 10K and 50K training data are presented in Table 5. Our results on full datasets are on par with baseline results and show improvements with fewer training data. With thousands of training samples, evaluation on FID is sufficient for this experiment.
Table 5. FID evaluation (lower is better) on CIFAR-10 and CIFAR-100 with BigGAN architecture.

| Method   | CIFAR-10 images | CIFAR-100 images |
|----------|----------------|-----------------|
|          | 5K             | 10K             | 50K          | 5K             | 10K             | 50K           |
| Baseline | 26.12          | 14.94           | 8.54         | 36.28          | 23.39           | 12.43         |
| Ours     | 20.83          | 12.75           | 8.60         | 33.94          | 21.97           | 12.20         |

Table 5. FID evaluation (lower is better) on CIFAR-10 and CIFAR-100 with BigGAN architecture.

For exploration we have tested implementing interpolation directly on original images, which led to sub-optimal solutions as expected. One can find the additional results in supplementary material. We have also tried using StyleGAN2 [22] backbone along with hinge loss as objective with multi-dimensional discriminator output, yet we did not find a satisfying baseline close to original results. This may be due to that different objective functions favor different parameter settings, while finding suitable hyper-parameters or regularization for hinge loss here is not trivial, which is also not the main consideration of this paper. In this case comparison with a weak baseline seems not necessary.

5. Discussion

In this paper we have proposed a simple yet effective implicit augmentation approach for low-shot data generation. Instead of producing new training images, the method functions in the multi-dimensional feature space of discriminator output. We further develop an algorithm which dynamically interpolates features by utilizing the characteristics of data itself during training. We also propose DScore as an additional metric to evaluate diversity of synthesized samples. Experiments show our method improves FID and DScore from strong baseline results on benchmark datasets, and can generate diverse appealing images with less than 100 training samples. Note that the method can synthesize new data which currently not exists. Whether it brings negative social impact may need further supervision.
References

[1] Relja Arandjelović and Andrew Zisserman. Three things everyone should know to improve object retrieval. In 2012 IEEE Conference on Computer Vision and Pattern Recognition, pages 2911–2918. IEEE, 2012.

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research, pages 214–223. PMLR, 06–11 Aug 2017.

[3] Randall Balestriero, Jerome Pesenti, and Yann LeCun. Learning in high dimension always amounts to extrapolation, 2021.

[4] Shane Barratt and Rishi Sharma. A note on the inception score. arXiv preprint arXiv:1801.01973, 2018.

[5] Ali Borji. Pros and cons of gan evaluation measures. ArXiv, abs/1802.03446, 2019.

[6] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In International Conference on Learning Representations, 2019.

[7] Junsuk Choe, Song Park, Kyungmin Kim, Joo Hyun Park, Dongseob Kim, and Hyunjung Shim. Face generation for low-shot learning using generative adversarial networks. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops, Oct 2017.

[8] Min Jin Chong and David Forsyth. Effectively unbiased fid and inception score and where to find them. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020.

[9] Ondřej Chum, Andrej Mikulík, Michal Perdoch, and Jiří Matas. Total recall ii: Query expansion revisited. In CVPR 2011, pages 889–896. IEEE, 2011.

[10] Ondřej Chum, James Philbin, Josef Sivic, Michael Isard, and Andrew Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In 2007 IEEE 11th International Conference on Computer Vision, pages 1–8. IEEE, 2007.

[11] Mengyu Dai and Haibin Hang. Manifold matching via deep metric learning for generative modeling. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pages 6587–6597, October 2021.

[12] Terrance DeVries and Graham W Taylor. Dataset augmentation in feature space. arXiv preprint arXiv:1702.05538, 2017.

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Advances in Neural Information Processing Systems, volume 27, 2014.

[14] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In Advances in neural information processing systems, 30, 2017.

[15] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In Advances in Neural Information Processing Systems, volume 30, 2017.

[16] Jie Hong, Pengfei Fang, Weihao Li, Tong Zhang, Christian Simon, Mehrtash Harandi, and Lars Petersson. Reinforced attention for few-shot learning and beyond. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 913–923, June 2021.

[17] Yan Hong, Li Niu, Jianfu Zhang, Jing Liang, and Liqing Zhang. Deltagan: Towards diverse few-shot image generation with sample-specific delta. arXiv preprint arXiv:2009.08753, 2020.

[18] Yan Hong, Li Niu, Jianfu Zhang, Weijie Zhao, Chen Fu, and Liqing Zhang. F2gan: Fusing-and-filling gan for few-shot image generation. In Proceedings of the 28th ACM International Conference on Multimedia, pages 2535–2543, 2020.

[19] Kai Huang, Jie Geng, Wen Jiang, Xinyang Deng, and Zhe Xu. Pseudo-loss confidence metric for semi-supervised few-shot learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pages 8671–8680, October 2021.

[20] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In Proc. NeurIPS, 2020.

[21] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.

[22] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In Proc. CVPR, 2020.

[23] JangHyun Kim, Wonho Choo, Hosan Jeong, and Hyun Oh Song. Co-mixup: Saliency guided joint mixup with super-modular diversity. In International Conference on Learning Representations, 2021.

[24] Jang-Hyun Kim, Wonho Choo, and Hyun Oh Song. Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In International Conference on Machine Learning (ICML), 2020.

[25] Byungsoo Ko and Geoimmo Gu. Embedding expansion: Augmentation in embedding space for deep metric learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2020.

[26] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).

[27] Yijun Li, Richard Zhang, Jingwan Lu, and Eli Shechtman. Few-shot image generation with elastic weight consolidation. arXiv preprint arXiv:2012.02780, 2020.

[28] Yiting Li, Haiyue Zhu, Yu Cheng, Wenhao Wang, Chek Sing Teo, Cheng Xiang, Prabhat Vadakkepat, and Tong Heng Lee. Few-shot object detection via classification refinement and distractor retreatment. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 15395–15403, June 2021.

[29] Jae Hyun Lim and Jong Chul Ye. Geometric gan, 2017.
[30] Bingchen Liu, Yizhe Zhu, Kunpeng Song, and Ahmed Elgammal. Towards faster and stabilized {gan} training for high-fidelity few-shot image synthesis. In *International Conference on Learning Representations*, 2021. 1, 2, 3, 5, 7, 8, 11

[31] Sihan Liu and Yue Wang. Few-shot learning with online self-distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 1067–1070, October 2021. 1

[32] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29:2234–2242, 2016. 5

[33] Hang Shao, Abhishek Kumar, and P. Thomas Fletcher. The riemannian geometry of deep generative models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018. 2, 3, 4

[34] Zhangzhang Si and Song-Chun Zhu. Learning hybrid image templates (hit) by information projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34:1354–1367, 2012. 5

[35] Stefan Stojanov, Anh Thai, and James M. Rehg. 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)*, pages 1798–1808, June 2021. 1

[36] Sunil Thulasidasan, Gopinath Chennupati, Jeff A Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 2

[37] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B. Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? *ArXiv*, abs/2003.11539, 2020. 1

[38] Han Zhang, Zizhian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient gan training. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020. 1, 2, 5, 11

[39] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Younghoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 2

[40] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 5

[41] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. 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*, pages 2109–2116. IEEE, 2009. 2

[42] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *International Conference on Machine Learning*, pages 6438–6447. PMLR, 2019. 2

[43] Huaxiu Yao, Xin Song, Zhiqiang Tao, Yaliang Li, Bolin Deng, Ruirui Li, and Zhenhui Li. Automated relational meta-learning. In *International Conference on Learning Representations*, 2020. 1

[44] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *CoRR*, 2015. 5
A. More Discussions and Experiments Using Direct Interpolation in Image Space

It is not obvious to adopt the proposed method in image space directly. 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. Moreover, even if one uses learned features to assign weights for interpolation, such weights derived from features cannot be directly used on original images. For example, given two valid features \( y_1 = g_w(x_1) \) and \( y_2 = g_w(x_2) \) where \( x_1, x_2 \) are real samples, using the proposed method we have \( \tilde{y} = \pi_1 y_1 + \pi_2 y_2 \) for some \( \pi_1, \pi_2 \). It is not equivalent to using \( \tilde{x} = \pi_1 x_1 + \pi_2 x_2 \) as interpolation on images, or in other words, \( \tilde{y} \neq g_w(\tilde{x}) \) here (as illustrated in Figure 1 in our paper). Hence, we simply tried implementing interpolation on images in a mixup style without labels. Same as implementing feature interpolation, we find \( p = 1 \) led to unsuccessful training sessions, thus we set \( p = 0.9 \) where generated samples seemed more reasonable. Table 6 shows FID and DScore evaluation results on \( 1024 \times 1024 \) experiments. One can compare it with Table 2 in the paper and the conclusion is obvious.

B. Implementation Details

Experiments with network architectures in [30]: We used Adam optimizer with learning rate \( 2 \times 10^{-4} \), \( \beta_1 = 0.5 \) and \( \beta_2 = 0.999 \) for both networks. Noise dimension was fixed to 256 and number of channels for both networks were set to 64. Batch size and \( k \) were set to 8 by default in all experiments. As to final evaluations, we used \( r = 0.2, s = -0.1 \) in experiments on Art and Shells, and \( r = 1, s = 0 \) on the rest of the datasets. For training sets with less than or equal to 300 samples we generated 1K images for evaluation; Otherwise we used 5K synthesized images. Experiments with network architectures in [6]: We used the training settings in [48] by default. For simplicity we set \( k = 2, t = 1 \) and batch size 16. The augmentation probability \( p \) was set to 0.5, 0.3, 0.1 in 5K, 10K, 50K experiments, respectively.
| Dataset     | Art  | Fauvism | Anime Face | Cat  | Dog   | Obama | Panda | Grumpy Cat | Shells |
|-------------|------|---------|------------|------|-------|-------|-------|------------|--------|
| Best FID    | 149.09 | 193.03  | 65.37      | 41.42 | 101.23 | 43.00 | 9.94  | 27.39      | 140.90 |
| DScore@Best | 0.0884 | 0.0938  | 0.0953     | 0.0610 | 0.0802 | 0.0591 | 0.0795 | 0.1001     | 0.0488 |

Table 6. Evaluation of generated samples using direct interpolation on images.