ON PREDICTING GENERALIZATION USING GANS

Anonymous authors
Paper under double-blind review

ABSTRACT

Research on generalization bounds for deep networks seeks to give ways to predict test error using just the training dataset and the network parameters. While generalization bounds can give many insights about architecture design, training algorithms etc., what they do not currently do is yield good predictions for actual test error. A recently introduced Predicting Generalization in Deep Learning competition (Jiang et al., 2020) aims to encourage discovery of methods to better predict test error. The current paper investigates a simple idea: can test error be predicted using *synthetic data,* produced using a Generative Adversarial Network (GAN) that was trained on the same training dataset? Upon investigating several GAN models and architectures, we find that this turns out to be the case. In fact, using GANs pre-trained on standard datasets, the test error can be predicted without requiring any additional hyper-parameter tuning.

This result is surprising because GANs have well-known limitations (e.g. mode collapse) and are known to not learn the data distribution accurately. Yet the generated samples are good enough to substitute for test data. Several additional experiments are presented to explore reasons why GANs do well at this task.

1 INTRODUCTION

Why do vastly overparametrized neural networks achieve impressive generalization performance across many domains, with very limited capacity control during training? Despite some promising initial research, the mechanism behind generalization remains poorly understood. A host of papers have tried to adapt classical generalization theory to prove upper bounds of the following form on the difference between training and test error

$$\text{test} - \text{train} \leq \sqrt{\frac{C}{|S|}} + \text{tiny term} \tag{1}$$

where S is the training dataset and C is a so-called *complexity measure*, typically involving some function of the training dataset as well as the net parameters (e.g., a geometric norm). Current upper bounds of this type are loose, and even vacuous. There is evidence that such classically derived bounds may be too loose (Dziugaite & Roy) or that they may not correlate well with generalization (Jiang et al. (2019)).

This has motivated more principled empirical study of effectiveness of generalization bounds. The general idea is to use machine learning to determine which network characteristics promote good generalization in practice and which do not —in other words, treat various deep net characteristics/norms/margins etc. as inputs to a machine learning model that uses them to predict the generalization error achieved by the net. This could help direct theorists to new types of complexity measures and motivate new theory.

A recently started competition Predicting Generalization in Deep Learning (Jiang et al., 2020) (PGDL) seeks to increase interest in such investigations, in an effort to uncover new and promising network characteristics and/or measures of network complexity that correlate with good generalization. As required in Jiang et al. (2020), a complexity measure should depend on the trained model, optimizer, and training set, but not the held out test data. The first PGDL competition in 2020 did uncover quite a few measures that seem to be predictive of good generalization but had not been identified by theory work.

In this paper, we explore a very simple baseline for predicting generalization that had hitherto not received attention: train a Generative Adversarial Network (GAN) on the training dataset, and use performance on the synthetic data produced by the GAN to predict generalization.

At first sight GANs may not appear to be an obvious choice for this task, due to their well known limitations. For instance, while the goal of GANs training is to find a generator that fools the discriminator net —in the sense that the discriminator has low probability of spotting a difference between GAN samples and the samples from the true distribution—in practice the discriminator is often able to discriminate very well at the end, demonstrating that it was not fooled. Also, GANs' generators are known to exhibit mode collapse i.e., the generated distribution is a tiny subset of the true distribution. There is theory and experiments suggesting this may be difficult to avoid (Arora et al., 2018). Along the same lines, Santurkar et al. (2018) and Bau et al. (2019) find that synthetic data distributions from GANs cannot match the distributional statistics of the training data. Ravuri & Vinyals (2019) showed that synthetic data from GANs is a poor substitute for real data when it comes to training classifiers.

Given all these negative results about GANs, the surprising finding in the current paper is that GANs *do* allow for good estimates of test error (and generalization error). This is verified for families of well-known GANs and datasets including primarily CIFAR-10/100, Tiny ImageNet and well-known deep neural classifier architectures. In particular we also tested on the PGDL and DEMOGEN benchmarks of predicting generalization and were able to do well on them (Section 3.1 and 3.2). We also investigate reasons behind the surprising efficacy of GANs in predicting generalization (Section 4) as well as the effects of the use of data augmentation during GAN training on predicting generalization.

2 Related Work

Generalization Bounds Traditional approaches to predict generalization construct a generalization bound based on some notion of capacity such as parameter count, VC dimension, Rademacher complexity, etc. Neyshabur et al. (2018) provide a tighter generalization bound that decreases with increasing number of hidden units. Dziugaite & Roy reveal a way to compute nonvacuous PAC-Bayes generalization bounds. Recently, bounds based on knowledge distillation have also come to light Hsu et al. (2020). Despite progress in these approaches, a study conducted by Jiang et al. (2019) with extensive hyper-parameter search showed that current generalization bounds may not be effective, and the root cause of generalization remains elusive. Given the arduous nature of constructing such bounds, an interest in complexity measures has arisen.

Predicting Generalization in Deep Learning The PGDL competition (Jiang et al., 2020) was held in NeuRIPS 2020 in an effort to encourage the discovery of empirical generalization measures following the seminal work of Jiang et al. (2018). The winner of the PGDL competition Natekar & Sharma (2020) investigated properties of representations in intermediate layers to predict generalization. Specifically, they investigate within class clustering, perturbed margin, and mixup classification. Another approach that performed well on PGDL exmained the effect of data augmentations on model accuracy. Kashyap et al. (2021), the second place winner, experiment with robustness to flips, random erasing, random saturation, and other such natural augmentations. Data augmentation, like GANs also represents a function of the original training data that produces data points not identical to the training data. However, the use of GANs eliminates the need to examine hand-selected transformations, like random erasure etc. After the competition, Jiang et al. (2021) interestingly find that generalization can be predicted by running SGD on the same architecture multiple times, and measuring the disagreement ratio between the different resulting networks on an unlabeled test set.

Criticism of PGDL There are some pitfalls in predicting generalization, as highlighted by Dziugaite et al. (2020). They find that distributional robustness over a family of environments is more applicable to neural networks than straight averaging.

Limitations of GANs Arora et al. (2018) introduce the Birthday Paradox test to conclude that GANs learn a distribution with relatively small support. Their work also reveals that GANs generate distinct images from their nearest neighbours in the training set. Santurkar et al. (2018) investigate the extent to which GAN generated samples can match the distributional statistics of the original

training data, and find that they have significant shortcomings. Specifically, they find that classifiers trained on GAN generated data have significantly lower accuracy on the true test data, and that the decision boundary on GAN data is simpler, i.e. better fit by a linear model than the decision boundary of true data. Ravuri & Vinyals (2019) find that GAN data is of limited use in training ResNet models, and find that neither inception score nor FID are predictive of generalization performance.

GANs do not simply memorize training data Initial attempts to detect whether GANs memorize the training set plotted GAN samples against their nearest neighbours in the test set. Webster et al. (2019) demonstrate that this test is too simplistic, but use latent recovery to conclude more carefully that GANs do not memorize the training set.

Dataset distillation Dataset distillation (Wang et al., 2018; Bohdal et al., 2020; Nguyen et al., 2020; Zhao & Bilen, 2021) aims to construct a dataset that is more informative than its original counterpart yet smaller in size. These ideas are of primary use in the training phase, as opposed to predicting generalization.

2.1 Notations and GAN architectures

Throughout the paper, we denote by $S_{\text{train}}, S_{\text{test}}$ and S_{syn} the training set, test set and the synthetic dataset generated by GANs. Given a classifier f, we write $g(f) := \frac{1}{|S_{\text{test}}|}\sum_{(x,y)\in S_{\text{test}}} \mathbf{1}\left[f(x)=y\right]$ and $\hat{g}(f) := \frac{1}{|S_{\text{syn}}|}\sum_{(\tilde{x},\tilde{y})\in S_{\text{syn}}} \mathbf{1}\left[f(\tilde{x})=\tilde{y}\right]$ to denote the test accuracy and synthetic accuracy of f.

In experiments, we extensively use the StudioGAN library by Kang & Park (2020) which provides default hyper-parameters along with pre-trained weights for state-of-the-art GAN models on standard datasets such as CIFAR-10 and Tiny ImageNet.

We investigate BigGAN, ContraGAN (Kang & Park, 2020), SA-GAN Zhang et al. (2019) and SN-GAN (Miyato et al., 2018). The original version of BigGAN appeared in Brock et al. (2019), and however we use the modified implementation from Kang & Park (2020). We also investigate variants such as BigGAN with adaptive discriminator augmentation (BigGAN-ADA) Karras et al. (2020), BigGAN with differentiable augmentation (BigGAN-DiffAug) Zhao et al. (2020). All GANs used in our experiments are conditional, i.e. they generate labeled examples.

3 Predicting test performance using GAN samples

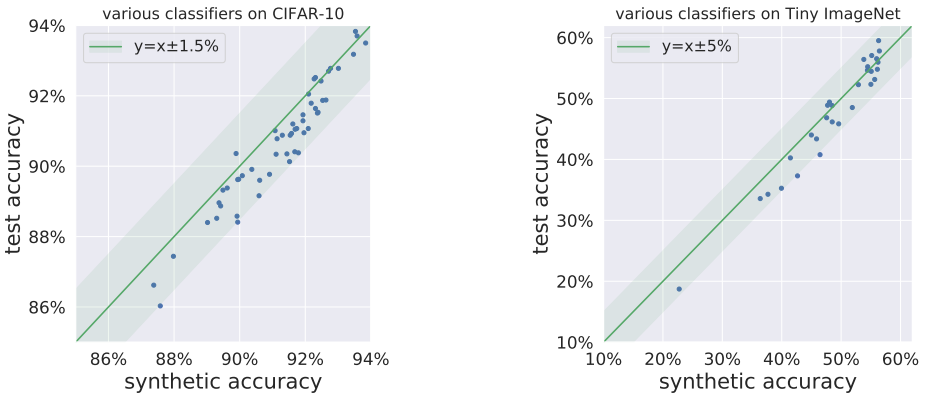


Figure 1: Scatter plots of test accuracy $g(f)$ v.s. synthetic accuracy $\hat{g}(f)$ with f from a pool of deep net classifiers on CIFAR-10 and Tiny ImageNet of diverse architectures VGG, ResNet, DenseNet, ShuffleNet, NASNet, MobileNet with various hyper-parameters. One single BigGAN+DiffAug model is used for each dataset. **Left**: CIFAR-10 classifiers. The $y = x$ fit has R^2 score 0.804 and Kendall τ coefficient 0.851. **Right**: Tiny ImageNet classifiers. The $y = x$ fit has R^2 score 0.918 and Kendall τ coefficient 0.803.

We now precisely define what it means to predict test performance in our setting. Given a classifier f trained on the training set S_{train}, we aim to predict its classification accuracy $g(f) := \frac{1}{|S_{\text{test}}|}\sum_{(x,y)\in S_{\text{test}}} \mathbf{1}\left[f(x) = y\right]$ on a hold-out test set S_{test}. Our proposal is to first train a *conditional* GAN model on the very same training set S_{train}, whose generator synthesizes *labeled* examples and then sample from the generator to construct a set of synthetic examples S_{syn}. In the end, we simply use f's performance on the synthetic dataset as our prediction for its test accuracy. Algorithm 1 formally describes this procedure.

Algorithm 1 Predicting test performance

Require: target classifier f, training set S_{train}, GAN training algorithm $\mathcal{A}$

1. Train a conditional GAN model using S_{train}:

$$G, D = \mathcal{A}(S_{\text{train}}) \text{ where } G, D \text{ are the generator and discriminator networks.}$$

2. Generate a synthetic dataset by sampling from the generator G:

$$S_{\text{syn}} = \{(\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_N, \tilde{y}_N)\} \text{ where } \tilde{x}_i, \tilde{y}_i = G(z_i, \tilde{y}_i).$$

The z_i's are drawn i.i.d. from G's default input distribution, and the $\tilde{y}_i$'s are chosen so as to match the empirical label distribution of the training set.

Output: the synthetic accuracy $\hat{g}(f) := \frac{1}{|S_{\text{syn}}|}\sum_{(\tilde{x},\tilde{y})\in S_{\text{syn}}} \mathbf{1}\left[f(\tilde{x}) = \tilde{y}\right]$ as the prediction

We demonstrate in Figure 1 that the test accuracy $g(f)$ consistently resides in a small neighborhood of $\hat{g}(f)$ for a diverse class of deep neural net classifiers trained on different datasets. To run Algorithm 1, we adopt the BigGAN model with the differentiable augmentation technique. We evaluate on pretrained deep neural net classifiers with architectures ranging from VGG-(11, 13, 19), ResNet-(18, 34, 50), DenseNet-(121, 169), ShuffleNet, PNASNet and MobileNet trained on CIFAR-10 and Tiny ImageNet with hyper-parameter setting `(learning rate, weight decay factor, learning rate schedule)` uniformly sampled from the grid $\left\{10^{-1}, 10^{-2}\right\} \times \left\{5 \times 10^{-4}, 10^{-3}\right\} \times \{\texttt{CosineAnnealing}, \texttt{ExponentialDecay}\}$.

In fact, it suffices to take the pre-trained BigGAN+DiffAug models on CIFAR-10 and Tiny ImageNet from the StudioGAN library, and sample exactly once for each datasets to generate the synthetic datasets for all predictions.

3.1 Evaluation on PGDL tasks

We evaluate our proposed method on the tasks of NeurIPS 2020 Competition on Predicting Generalization of Deep Learning and compare to the solutions from the Top-3 winning teams. The PGDL competition consists of 8 tasks[1], and each task contains a set of pre-trained deep net classifiers of similar architectures (mostly VGGs and Network in Networks) but with different hyper-parameter settings (width, depth, learning rate etc.) as well as the training data they are trained on.

For each task, the goal is to compute a scalar prediction for each classifier based on its parameters and the training data that *correlates* as much as possible with the actual generalization errors of the classifiers measured on a held-out test set. The correlation score is measured by the so-called Conditional Mutual Information, which is designed to indicate whether the computed predictions contain all the information about the generalization errors such that knowing the hyper-parameters does not provide additional information, see (Jiang et al., 2020) for details. The solution obtaining the highest average score over the hidden tasks (Task 6, 7, 8, 9) gets the first place.
Our proposed solution for each PGDL task is that we first train a GAN model on the provided training dataset and sample from it a labeled synthetic dataset, and then for each classifier, we simply use the gap between its given training accuracy and its accuracy measured on a synthetic dataset . Specifically, we adopt a BigGAN model with the default hyper-parameters for CIFAR-10.[2]

[1] namely Task 1, 2, 4, 5, 6, 7, 8, 9, without 3.

[2] we take the default hyper-parameters for Task 1, 8 and 9, and slightly modify the hidden dimensions of so it runs on the 28×28 Fashion-MNIST dataset for Task 2.

Task	No.1 team			No.2 team	No.3 team	**Ours**
	DBI*LWM	MM	AM	R2A	VPM	
1 : VGG on CIFAR-10	25.22	1.11	15.66	52.59	6.07	**62.62**
2 : NIN on SVHN	22.19	47.33	**48.34**	20.62	6.44	28.57
8 : VGG on F-MNIST	9.24	1.48	1.28	13.79	16.23	**30.25**
9 : NIN on CIFAR-10	25.86	20.78	15.25	11.30	2.28	**29.27**

Table 1: Comparison of our method to the Top-3 teams of the PGDL competition. Scores shown are calculated on the Conditional Mutual Information metric (higher is better). DBI*LWM=Davies-Bouldin Index*Label-wise Margin, MM=Mixup Margin and AM=Augment Margin are the proposed methods from first place solution by Natekar & Sharma (2020). R2A=Robustness to Augmentations and VPM=Variation of the Penultimate layer with Mixup are the second and third place solutions by Aithal et al. (2021) and Lassance et al. (2020) respectively.

The tasks of PGDL are based on a wide range of datasets inlcuding CIFAR-10, SVHN, CINIC-10, Oxford Flowers, Oxford Pets and Fashion MNIST, and we found that a subset of them were not well-suited for standard GAN training without much tweaking, either owing to small training set or to highly uneven class sizes. We focused only on the subset of the tasks where GAN training worked out of the box, specifically Task 1, 2, 8, and 9. The results are shown in Table 1. We observe that on tasks involving either CIFAR-10 or VGG-like classifiers, our proposed solution out-performs any of the Top-3 teams's solutions of the competition by a large margin. One potential reason may be that the default hyper-parameters of the BigGAN model have been engineered towards CIFAR-10 like datasets and VGG/ResNet-like discriminator networks. It is noteworthy to mention we conducted absolutely zero hyper-parameter search in these experiments for PGDL, and we leave fine-tuning the GAN hyper-parameters and training procedure for each PGDL task to future work.

3.2 Evaluation on DEMOGEN benchmark

The seminal work of Jiang et al. (2018) constructed the Deep Model Generalization dataset (DEMOGEN) that consists of 216 ResNet-32 (He et al., 2016) models and 216 Network-in-Networks (Lin et al., 2013) models trained on CIFAR-10 plus another 324 ResNet-32 models trained on CIFAR-100, along with their training and test performance. These models are configured with different regularization techniques (group-norm v.s. batch-norm), widths, learning rates, weight decay factors, and notably whether to train with data augmentation.

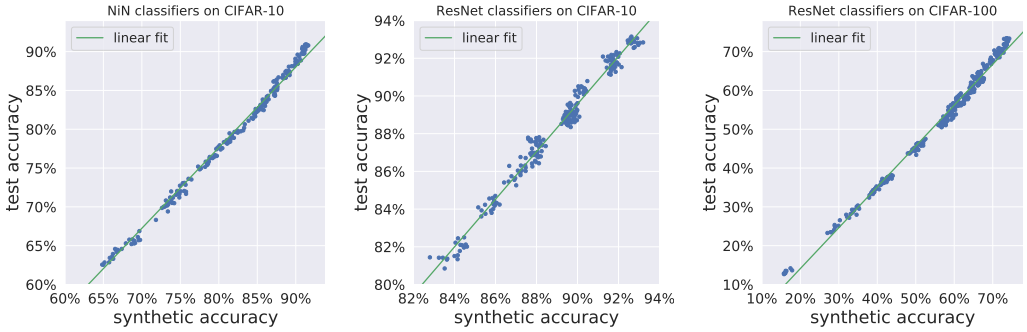


Figure 2: Scatter plots of test accuracy $g(f)$ v.s. synthetic accuracy $\hat{g}(f)$ with f from DEMOGEN consisting of 216 Network in Network classifiers and 216 ResNet classifiers on CIFAR-10 as well as 324 ResNet classifiers on CIFAR-100. The synthetic datasets are sampled from the BigGAN-DiffAug models listed in Table 2.

The goal of DEMOGEN, different from PGDL, is to numerically predict generalization gap (or equivalently test accuracy) with the additional knowledge of the test accuracies of a few pre-trained classifiers. Following the setup of Jiang et al. (2018), we use a simple linear predictor $g(f) = a \cdot \hat{g}(f) + b$ with two scalar variables $a, b \in \mathbb{R}$ instead of using $\hat{g}(f)$ directly for the bets

	NiN+CIFAR-10		ResNet+CIFAR-10		ResNet+CIFAR-100	
	adj R^2	10-fold R^2	adj R^2	10-fold R^2	adj R^2	10-fold R^2
`qrt+log`	0.94	0.90	0.87	0.81	0.97	0.96
`qrt+log+unnorm`	0.89	0.86	0.82	0.74	0.95	0.94
`moment+log`	0.93	0.87	0.83	0.74	0.94	0.92
Self-Attention GAN	0.992	0.991	0.903	0.897	—	—
Spec-Norm GAN	0.994	0.993	0.874	0.869	—	—
BigGAN	0.988	0.986	0.911	0.904	0.992	0.991
BigGAN-ADA	0.991	0.989	0.932	0.928	—	—
BigGAN-DiffAug	0.995	0.995	0.981	0.921	0.993	0.993

Table 2: Comparison of predicting generalization using GAN samples to the Top-3 methods of Jiang et al. (2018) (`qrt+log`, `qrt+log+unnorm` and `moment+log`) on DEMOGEN. Overall, our method of predicting generalization using GAN samples outperforms any of the previous solution by a large margin. The BigGAN model trained with differentiable augmentation appears to be the best performing. '—' means not tested.

performance. Here a, b are estimated using least squares regression on a pool of pre-trained classifiers independent of f with known test accuracies. Jiang et al. (2018) proposes to use variants of coefficient of determination (R^2) of the resulted predictor as evaluation metrics. Specifically, the adjusted R^2 and k-fold R^2 are adopted (see Jiang et al. (2018) for concrete definitions).

In Table 2, we report the scores of our method with several state-of-the-art GAN models. Specifically, again we take the pre-trained GAN models from the StudioGAN library to generate synthetic datasets for the CIFAR-10 tasks. However, since no CIFAR-100 GAN model is provided by the library, we use the hyper-parameters of the CIFAR-10 counterparts unmodified and train a BigGAN baseline and a BigGAN-DiffAug model from scratch, from which synthetic datasets are sampled. Again, each synthetic dataset is sampled only once. Overall, predicting using GAN samples achieves remarkably higher scores than any method proposed in Jiang et al. (2018). We also visualize the resulted linear fit for each task in Figure 2, where the strong linear correlation is well-observed. Note that the resulted linear fits are not too off from the ideally desired $y = x$ fit. In fact, the $y = x$ fit, i.e. using directly $\hat{g}(f)$ as the prediction, would result in extremely competitive R^2 values 0.928, 0.995 and 0.975 as well for each task.

4 GAN SAMPLES ARE CLOSER TO TEST SET THAN TRAINING SET

In this section, we demonstrate a surprising property of the synthetic examples produced by GANs that may provide insights into their observed ability to predict generalization: *Synthetic datasets generated by GANs are 'closer' to test set than training set in the feature space of well-trained deep net classifiers, even though GAN models had (indirect) access to the training set but no access to the test set*.

Here 'well-trained' classifiers are the ones perfectly classifying the training examples, i.e. its training accuracy is $> 97\%$. To define closeness properly, we introduce our class-conditional Fréchet distance $d_h(S, \tilde{S})$ between two sets S and $\tilde{S}$ of labeled examples, which builds upon a widely used standard evaluation metric for GANs, the Fréchet Inception Distance (FID) proposed by Heusel et al. (2017).

4.1 CLASS-CONDITIONAL FRÉCHET DISTANCE

The goal is to measure how close the *labeled* synthetic dataset is to the test set in the eyes of a certain pretrained classifier. However, FID does not capture any label information by default. Our proposed class-conditional Fréchet Distance can be viewed as the sum over each class of the Fréchet Distance in a certain feature space between test and synthetic examples. Originally, FID measures

Figure 3: Histograms of d_h's measured using the feature spaces of the pool of 216 pre-trained ResNet models on CIFAR-10 in DEMOGEN. Since features vectors $h(x)$ of different classifiers may have vastly different norms, we plot the histograms of the ratios $d_h(S_{\text{syn}}, S_{\text{test}})/d_h(S_{\text{train}}, S_{\text{test}})$ and $d_h(S_{\text{syn}}, S_{\text{test}})/d_h(S_{\text{syn}}, S_{\text{train}})$ instead. The synthetic dataset is the identical one generated by the BigGAN-DiffAug model used in Section 3.2. **Top**: the histograms of all the 216 pre-trained classifiers, including the not-so-well-trained ones with training accuracy below 90%. **Bottom**: the histograms of only the well-trained 162 out of 216 classifiers with training accuracy at least 97%. Clearly, the ratio $d_h(S_{\text{syn}}, S_{\text{test}})/d_h(S_{\text{train}}, S_{\text{test}})$ is below 1 consistently, and the ratio $d_h(S_{\text{syn}}, S_{\text{test}})/d_h(S_{\text{train}}, S_{\text{test}})$ is below 1 for all well-trained classifiers. This indicates that in their feature space the labeled synthetic examples behave more similarly to the test set rather than to the training set.

a certain distance between the feature embedding of the real examples and the synthetic examples from GANs in the feature space of an Inception network pretrained on ImageNet. For our purpose, we generalize the notion of FID and define the Fréchet distance between any two sets of examples S and S' w.r.t. any arbitrary feature extractor[3] h as

$$\mathcal{F}_h(S, S') := \|\mu_S(h) - \mu_{S'}(h)\|_2^2 + \operatorname{Tr}\left[\operatorname{Cov}_S(h) + \operatorname{Cov}_{S'}(h) - 2\left[\operatorname{Cov}_S(h_f)\operatorname{Cov}_S(h_f)\right]^{1/2}\right]$$

where we have the empirical mean of h's output $\mu_S(h) := \frac{1}{|S|}\sum_{(x,y)\in S} h(x)$ and the empirical covariance $\operatorname{Cov}_S(h) := \frac{1}{|S|}\sum_{(x,y)\in S}(h(x) - \mu_S(h))(h(x) - \mu_S(h))^\top$ over the examples from S, and $\mu_{S'}(h), \operatorname{Cov}_S(h)$ are computed analogously. Furthermore, $[\cdot]^{1/2}$ is the matrix square root and $\operatorname{Tr}[\cdot]$ is the trace function.

For labeled datasets S and $\tilde{S}$, we denote by $\mathcal{C}$ the collection of all possible labels. For each label $c \in \mathcal{C}$, we let S_c and $\tilde{S}_c$ denote the the subsets of examples with label c in S and $\tilde{S}$ respectively. This

[3]If h is the pool-3 layer output of the Inception net, $\mathcal{F}_h$ is identical to FID.

leads to the class-conditional Fréchet Distance:

$$d_h\left(S, \tilde{S}\right) := \sum_{c \in \mathcal{C}} \mathcal{F}_h\left(S_c, \tilde{S}_c\right)$$

In Figure 3, we evaluate d_h on CIFAR-10 among the training set S_{train}, the test set S_{test} and the synthetic set S_{syn} for h being the feature map immediately before the fully-connected layer of each pre-trained deep net used in the middle sub-figure of Figure 2. The synthetic dataset is generated by the same BigGAN+DiffAug model used in Figure 2 as well. We observe that for any pre-trained net, $d_h(S_{\text{syn}}, S_{\text{test}}) < d_h(S_{\text{train}}, S_{\text{test}})$ by a margin, which provides a partial explanation for why GANs' synthesized examples may replace the test set. Furthermore, for every well-trained nets, $d_h(S_{\text{syn}}, S_{\text{test}}) < d_h(S_{\text{syn}}, S_{\text{train}})$ as well. This is surprising because S_{syn} ends up being 'closer' to the unseen S_{test} rather than S_{train}, despite the GAN model has never observed S_{test} and the generator is trained to minimize certain distance between S_{syn} and S_{train} as part of the training objective.

4.2 Class-conditional Fréchet Distance across diverse classifier architectures

It is natural to wonder whether we may attribute the phenomenon of GAN samples being closer to test set to the architectural similarity between the evaluated classifiers and the discriminator network used in the BigGAN model for the figure — they are all derivatives from the ResNet family. Thus we investigate the values of d_h for the collection of classifiers with diverse architectures used in Figure 1, which contains network architectures that may be remote from ResNet such as VGG, MobileNet, PNASNet etc. The result is demonstrated in Figure 4, where we observe the surprising phenomenon that the single copy of the synthetic dataset is regarded closer to the test set by all classifiers in the collection. Thus this suggests certain GAN models are capable of matching to a degree the feature distribution universally for a diverse set of deep neural classifiers.

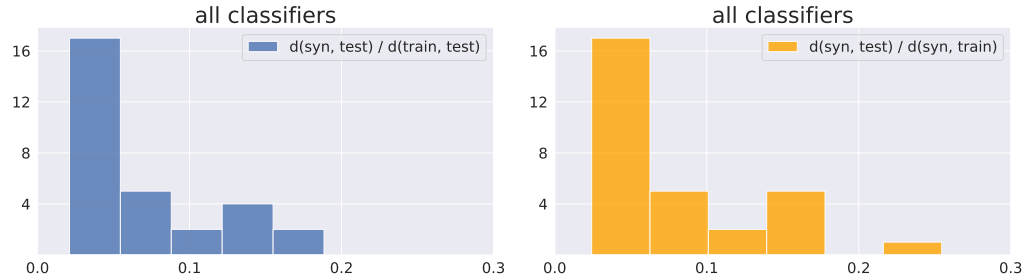
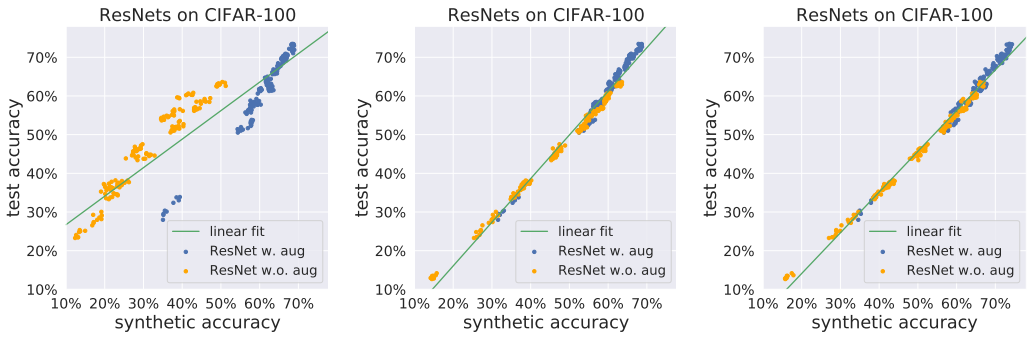


Figure 4: Histograms of d_h's measured using the feature spaces of deep neural nets with various architectures on CIFAR-10 (same ones from Figure 1). The synthetic dataset is the same one used in Figure 1 as well. All the classifiers are well-trained. $d_h(S_{\text{syn}}, S_{\text{test}})$ is consistently the smallest among all three pairwise distances among S_{train}, S_{test} and S_{syn}.

5 Exploration: Effects of data augmentation in GAN training on predicting generalization

In this section, we discuss certain intriguing effects of data augmentation techniques used during GAN training on the performance of the resulted synthetic datasets for predicting generalization. We use the CIFAR-100 task from DEMOGEN as a demonstration where the effects are the most pronounced. The task consists of 324 ResNet classifiers, only half of which are trained with data augmentation, specifically random crop and horizontal flipping.

Given the success of complexity measures that verify 'robustness to augmentation' in the PGDL competition, one would imagine applying data augmentations to the real examples during GAN training could be beneficial for prediciting generalization, since the generator would be encouraged to match the distribution of the augmented images. However, we show in Figure 5 this is not the case here.



(a) (b) (c)

Figure 5: Scatter plots of test accuracy $g(f)$ v.s. synthetic accuracy $\hat{g}(f)$ on the DEMOGEN CIFAR-100 task using BigGAN models trained with **a)** data augmentation (random crop + horizontal Flipping) only to real (training) examples, **b)** without any data augmentation, and **c)** with the differentiable data augmentation technique. Blue and orange dots represent ResNet-34 classifiers trained with and without data augmentation.

Specifically, Figure 5a indicates that, when evaluated on a synthetic dataset from a GAN model trained with augmentation only applied to real examples, the ResNets trained without any data augmentation exhibit a test accuracy v.s. synthetic accuracy curve that substantially deviates from $y = x$, while the ones trained with data augmentations are relatively unaffected. In comparison, once data augmentation is removed from GAN's training procedure, this deviation disappears, as captured in Figure 5b. This might be due to that the ResNets without data augmentation are sensitive to the perturbations learned by the generator. Overall, the best result is achieved by deploying the differential augmentation technique that applies the same differentiable augmentation to both real and fake images and enables the gradients to be propagated through the augmentation back to the generator so as to regularize the discriminator without manipulating the target distribution.

6 Conclusion

Competitions such as PGDL are an exciting way to inject new ideas into generalization theory. The interaction of GANs and good generalization needs further exploration. We leave it as an future work to determine the limits of using GANs for predicting generalization. It is conceivable that the quality of the GAN-based prediction is correlated with niceness of architectures in some way. It would not be surprising that it works for popular architectures but does not work for all. A theoretical understanding of many of the empirical phenomena presented in this paper could be a good first step in this direction.

In the line of analyzing GANs' functionality, previous work has shown various limitations of GANs and has questioned the utility of their samples. We hope our use of GAN samples to predict generalization highlights some positive use cases for GAN samples as well as sheds light on improving existing models and developing new theories for GANs.

References

Sumukh Aithal, Dhruva Kashyap, and Natarajan Subramanyam. Robustness to augmentations as a generalization metric. *arXiv preprint arXiv:2101.06459*, 2021.

Sanjeev Arora, Andrej Risteski, and Yi Zhang. Do gans learn the distribution? some theory and empirics. In *International Conference on Learning Representations*, 2018.

David Bau, Jun-Yan Zhu, Jonas Wulff, William Peebles, Hendrik Strobelt, Bolei Zhou, and Antonio Torralba. Seeing what a gan cannot generate. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

Ondrej Bohdal, Yongxin Yang, and Timothy Hospedales. Flexible dataset distillation: Learn labels instead of images. *arXiv e-prints*, pp. arXiv–2006, 2020.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1xsqj09Fm.

Gintare Karolina Dziugaite and Daniel M Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data.

Gintare Karolina Dziugaite, Alexandre Drouin, Brady Neal, Nitarshan Rajkumar, Ethan Caballero, Linbo Wang, Ioannis Mitliagkas, and Daniel M Roy. In search of robust measures of generalization. *Advances in Neural Information Processing Systems*, 33, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

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. *Advances in neural information processing systems*, 30, 2017.

Daniel Hsu, Ziwei Ji, Matus Telgarsky, and Lan Wang. Generalization bounds via distillation. In *International Conference on Learning Representations*, 2020.

Yiding Jiang, Dilip Krishnan, Hossein Mobahi, and Samy Bengio. Predicting the generalization gap in deep networks with margin distributions. *arXiv preprint arXiv:1810.00113*, 2018.

Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2019.

Yiding Jiang, Pierre Foret, Scott Yak, Daniel M Roy, Hossein Mobahi, Gintare Karolina Dziugaite, Samy Bengio, Suriya Gunasekar, Isabelle Guyon, and Behnam Neyshabur. Neurips 2020 competition: Predicting generalization in deep learning (version 1.1). *arXiv preprint arXiv:2012.07976*, 2020.

Yiding Jiang, Vaishnavh Nagarajan, Christina Baek, and J Zico Kolter. Assessing generalization of sgd via disagreement. *arXiv e-prints*, pp. arXiv–2106, 2021.

Minguk Kang and Jaesik Park. ContraGAN: Contrastive Learning for Conditional Image Generation. 2020.

Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. 2020.

Dhruva Kashyap, Natarajan Subramanyam, et al. Robustness to augmentations as a generalization metric. *arXiv preprint arXiv:2101.06459*, 2021.

Carlos Lassance, Louis Béthune, Myriam Bontonou, Mounia Hamidouche, and Vincent Gripon. Ranking deep learning generalization using label variation in latent geometry graphs. *arXiv preprint arXiv:2011.12737*, 2020.

Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

Parth Natekar and Manik Sharma. Representation based complexity measures for predicting generalization in deep learning, 2020.

Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. Towards understanding the role of over-parametrization in generalization of neural networks. 2018.

Timothy Nguyen, Zhourong Chen, and Jaehoon Lee. Dataset meta-learning from kernel ridge-regression. In *International Conference on Learning Representations*, 2020.

Suman Ravuri and Oriol Vinyals. Seeing is not necessarily believing: Limitations of biggans for data augmentation. 2019.

Shibani Santurkar, Ludwig Schmidt, and Aleksander Madry. A classification-based study of covariate shift in gan distributions. In *International Conference on Machine Learning*, pp. 4480–4489. PMLR, 2018.

Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

Ryan Webster, Julien Rabin, Loic Simon, and Frédéric Jurie. Detecting overfitting of deep generative networks via latent recovery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11273–11282, 2019.

Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International conference on machine learning*, pp. 7354–7363. PMLR, 2019.

Bo Zhao and Hakan Bilen. Dataset condensation with differentiable siamese augmentation. *arXiv e-prints*, pp. arXiv–2102, 2021.

Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient gan training. *Advances in Neural Information Processing Systems*, 33, 2020.