A Quantitative Measure for Generative Models with Applications to Generative Adversarial Networks

Abstract

Evaluating the quality of images has been a challenging task, and the importance of a quantitative measure has grown significantly with the recent popularity of Generative Adversarial Networks. An important consideration to make is that the model should generate a set of good images, and not only a single good example. Hence we considered distributions of images when constructing our Distribution Divergence Measure (DDM). The measure computes the relative divergence of a test set and generated samples, this approximates how well a model generalizes to the true distribution. We demonstrate its effectiveness as a quality measure on Generative Adversarial Networks.

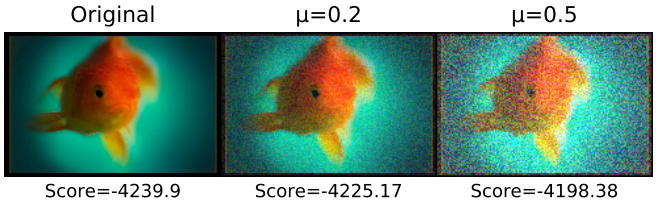


Figure 1: ImageNet results highlighting how increasing the level of corruption causes a worse score. This image represents a sample from a distribution of corrupted images where the first image has no corruption, the second and third image underwent uniform additive noise from $\texttt{Unif}[0, 0.2]$ and $\texttt{Unif}[0, 0.5]$, respectively. The score underneath represents the measures output for the entire distribution undergoing that corruption where a lower score indicates better performance.

1. Introduction

When Generative Adversarial Networks (GAN) researchers claim state of the art in image generation [12], they may inadvertently overstate their results due to a lack of standardized benchmarks. Worse, within many papers the final results shown are cherry-picked and often unrepresentative of the model, as there is often a sizable discrepancy between best case samples and average case samples. We exemplify how cherry-picked can mislead in Figure 3. Misleading results can hinder progress and limit adoption outside this research avenue.

Standardized measures can limit misleading results, though measures can also overstate their utility for model comparison. We discuss previous measures in Section 2.2 and explore their performance and pitfalls in Section 5.1. We will see that previous measures fail to capture differences between generative models, so we present a new measure that compares the distribution generated by a model with the test set distribution (Section 3).

We will cover three of the previous methods namely Parzen Window Estimation [14], Annealed Importance Sampling [20], and the Inception Score [18]. The first method is oldest with the other two representing relatively recent attempts to solve the issue. The first two are measures that estimate the log-likelihood of the data while the last method the Inception Score is a nonparametric measure that only relies on generated distribution to score the generative model. We feel that a nonparametric measure will ultimately be best suited for both adoption and as a valid measure of quality because the availability of tunable hyperparameters will allow for a measure to be compromised in favor of the model of interest rather than a true and fair measure of quality.

Besides requiring the measure to be nonparametric we also wanted to create a measure that can be evaluated on a finite set of samples and reflect how well the generative model matches the true distribution. By desiring to measure closeness to the true distribution we require a test set that the generative model does not have access to. We can use the test set as a proxy for how well the generator generalizes to the true distribution. Given these desiderata we arrive to computing the relative divergence between the generated samples and the test set.

2. Background

2.1. Generative Adversarial Networks

We briefly cover the original Generative Adversarial Network (GAN) [4] formulation before describing a few of the modifications that were made in the architectures we use.

For Generative Adversarial Networks, there are two competing networks a generator G and a discriminator D. The generator takes as input noise z from a distribution $\mathbb{P}_z$, typically Gaussian, and outputs an image. The generator tries to match the target distribution $\mathbb{P}_r$ typically taken to be natural images x. The output of the Discriminator is from $[0, 1]$ which discriminates between real and fake images by learning to assign a score of 0 to fake images and 1 to real images. The optimization is performed jointly on both networks optimizing the following loss.

$$\min_G \max_D L(D, G) = \mathbb{E}_{x\sim\mathbb{P}_r} \log(D(x)) + \mathbb{E}_{z\sim\mathbb{P}_z} \log(1 - D(G(z)))$$

The specific GANs we consider in this paper are that of DCGAN [15], Improved GAN [18], and Improved Wasserstein GAN [5]. The following descriptions are oversimplification of the changes from the initial GAN architecture. The DCGAN is an architectural modification in that they replaced the multilayer perceptron with a convolutional neural network (CNN). The Improved GAN builds from DCGAN by using a CNN and modifies the loss by penalizing the difference between layers of generator and discriminator. Finally the Improved Wasserstein GAN changes the loss to $\mathbb{E}_{x\sim\mathbb{P}_r}[D(x)] - \mathbb{E}_{z\sim\mathbb{P}_z}[D(G(z))]$ and applies a penalty to the gradients so that the CNN approximates a L_1 Lipschitz function.

2.2. Previous Measures

Parzen Window Estimation. [14] uses a kernel centered around each data point to compute the density function of the data. It belongs to a family of kernel methods that estimate the log likelihood of the data. The problem with this family of methods is that they fail to properly track image quality and can be manipulated to achieve a desired result. Moreover [19] demonstrate how image quality provides no information about likelihood and vice-versa. In addition the estimates from the methods can be quite far from the true likelihoods. We also demonstrate how Parzen Windows fails to track image quality in section 5.1.

Annealed Importance Sampling. [20] follows in a similar vein by trying to estimate the log-likelihood of the data. To estimate the log-likelihood in AIS, they consider the geometric mean of many intermediate distributions between the prior, assumed to be Gaussian with respect to the decoder model, and the target, the output of the decoder. However, in GANs there is no restriction or implicit constraint to its target distribution to be that of a Gaussian. The Gaussian prior assumption is met within variational autoencoders (VAE) by explicitly penalizing the model from deviating from a Gaussian code. This loss in VAEs explains why they score best using AIS as compared to GANs.

Still the main drawback to AIS was explained by [19] who show that log-likelihood based estimates are a poor measure. We further demonstrate how even within a fixed model the estimates can vary by many nats by changing the variance of the Gaussian prior in Figure 6.

The Inception Score. [18] uses the inception model which is trained on [17] and takes the form: $\exp(\mathbb{E}_x\left[\mathrm{KL}(p(y|x)\|p(y))\right])$. This measure might arguably be the most popular measure currently used to measure quality of a GAN. This measure was shown to correlate well with image quality assessment by Mechanical Turkers and also promotes diversity of samples.

There are several faults with the Inception Score namely: its internal bias, its discouragement intraclass diversity, and its failure to capture repeated modes. The first point attests to how the model used in the Inception Score is biased to images most similar to those found in ImageNet due to its training procedure and therefore not reliable for images that differ significantly from those found in ImageNet. There exist many natural datasets which differ significantly from ImageNet, such as SVHN [3], satellite imagery[13], and NORB[10]. This fact alone concerning the model's bias would not represent an issue, but the score does not take into account the target distribution so the score is only ever relative to the generated distribution. The second point can be observed by examining what increases the score most and those are images that activate a particular ImageNet class. Therefore the measure promotes activation of many ImageNet classes as opposed to activating a single or varied distribution of classes, which can be regarded as promoting interclass variation and penalizing for intraclass variation. The last point we demonstrate experimentally in section 5.5, but can also be observed as well in the formulation of the score. Where the score penalizes for only containing a small number of modes by way of the $p(y)$ being close to $p(x|y)$ if there are only a handful of modes. However once the number of modes exceeds some threshold $p(y)$ will approach the maximum entropy for a given dataset and then the measure itself becomes saturated.

A measure that we do not consider in this paper is classifier two sample test [11]. The method involves training a neural network to classify images as being real or fake, and then computing the average amount that it gets fooled to serve as the statistic. This method is akin to training a discriminator during the training of GANs. By relying on a neural network to classify in versus out of distribution there are no guarantees it will produce any meaningful ordering for two models which are both in or both out of distribution. As well the model would fail to detect the difference between an autoencoder and a generative model.

Figure 2: CIFAR-10 samples from Improved GAN without Batch Normalization and the normal Improved GAN. Images are randomly sampled during the end of training.

3. Measure

In this paper we aim to create a measure that is optimal if it matches the true distribution. Since the true distribution is unknown to us, we approximate the true distribution with a set of finite samples S drawn from the true distribution. Given S we can split the samples into a held out test set T and a training set R. Similar to other computer vision tasks such as classification we would like to measure how well the generator performs by evaluating the generator on the test distribution.

One approach to approximating images as distributions to consider each image as a delta function around which some mass exists. This creates a point-wise mass distribution that can be used in common for each distribution. Even with this representation though computing the actual probability distribution of space is still intractable. From here we further relax the distributions to be modeled as a Gaussian mixture. Computing the diverge of two Gaussian mixtures is costly in their general form but we can approximate them with a variational approximation which will provide us with a lower bound on the true divergence [2].

We arrive at our variational approximation by simplifying [2] work in approximating Gaussian mixtures. The following derivation begins from said work. We will use π and ω to correspond to the mixing weights of the Gaussian mixture.

The likelihood $L_f(g) = \mathbb{E}_{f(x)}[\log\, g(x)]$ relates to the KL divergence by $\mathrm{KL}[f\|g] = L_f(f) - L_f(g)$. Therefore any estimate of likelihood can be related back to the KL.

By Jensen's inequality we have

$$\begin{aligned} L_f(g) &= \mathbb{E}_{f(x)} \log\, g(x) \\ &= \mathbb{E}_{f(x)} \log \sum\, \omega_b g_b(x) \\ &= \mathbb{E}_{f(x)} \log \sum\, \phi_{b|a} \frac{\omega_b g(x)}{\phi_{b|a}} \\ &\geq \mathbb{E}_{f(x)} \sum\, \phi_{b|a} \log \frac{\omega_b g(x)}{\phi_{b|a}} \\ &= \mathcal{L}_f(g, \phi) \end{aligned} \tag{1}$$

This being a lower bound on $L_f(g)$, we get the best bound by maximizing $\mathcal{L}_f(g, \phi)$ with respect to ϕ.

$$\hat{\phi}_{b|a} = \frac{\omega_b e^{-\mathrm{KL}(f_a \| g_b)}}{\sum_{b'} \omega_{b'} e^{-\mathrm{KL}(f_a \| g_{b'})}} \tag{2}$$

A similar bound can be achieved for $L_f(f)$. Finally we define the variational approximation by substituting ϕ into $L_f(g)$ and the corresponding ψ into $L_f(f)$.

$$\mathrm{KL}(f\|g) = \sum_a \pi_a \log \frac{\sum_{a'} \pi_{a'} e^{-\mathrm{KL}(f_a \| f_{a'})}}{\sum_{b'} \omega_{b'} e^{-\mathrm{KL}(f_a \| g_b)}} \tag{3}$$

In our specific case we can simplify the variational approximation to what's below due to some assumptions. Namely we assume an equal weighting of each Gaussian in the mixture. Thus π and ω become a normalization parameter equal to the number of examples in each respective mixture. We can also ignore the numerator as in our case it is a constant it would only act as a linear shift to our results.

$$-\frac{1}{|T|} \log \left[\frac{1}{|G|} \sum_{g \in G} e^{-f(T,G)} \right] \tag{4}$$

Given this final formulation we can observe a few nice properties of the measure. It can achieve its optimal performance when the generator matches the test distribution.

An important note about the variational approximation is that it provides a lower bound for KL, however, the restriction of positivity is lost. Therefore the results presented in the paper are negative despite being presented as the KL.

3.1. Measure Embedding

In the previous section we derived the Distribution Divergence Measure as a lower bound of the KL Divergence between two Gaussian Mixtures. It still leaves open the question of which embedding to use when computing the measure, and if an embedding is even necessary.

The main reason to have the measure computed on an embedding rather than on the pixels themselves is twofold. The first reason is that by considering an embedding space that is much smaller than the original space we can avoid

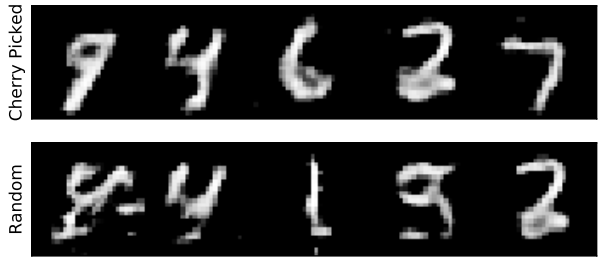


Figure 3: An example GAN on trained on MNIST showing that cherry-picked examples accrued from a GAN can be deceptive.

the curse of dimensionality. This also works to ensure long term use of the measure as images are increasingly growing in resolution so any measure that works directly in pixel space will increase its computation time. The second reason to utilize an embedding is that it can compress images into the core components of what is desired such as being natural. There are many factors such as slight changes in blur, brightness, or contrast that all allow an image to remain natural so the embedding should be immune to slight variations of these modifications.

These requirements lead us to a consider embeddings that are invariant to slight changes in brightness, contrast, or blur, of which a natural choice are neural networks. Neural Network models are currently insensitive to slight changes in all of the aforementioned criterion and can produce at above human level performance in some tasks such as that of classification. Given this we chose our embedding to be the logits of a classifier trained on the dataset of interest. The requirement of a pretrained classifier is currently a non-issue due to the wide availability of pretrained models on most if not all of the standard datasets of which GANs are trained on.

Logits. are the final output of a neural network before taking the softmax. The logits represent a natural choice as they are the most compact dimensional representation while still containing semantic information. The main reason to choose logits over the final output probabilities is that [7] has shown that the logits contain a lot of information that is lost after the softmax. The logits themselves can be used to train other networks [16].

3.2. Alternative Forms

Besides the form of the measure we have presented in equation 4 we also considered two alternative forms of the measure. For the first alteration we considered the relative relative divergence two pairs of distributions train R and test T, and train and generated G. The formulation is presented below in equation 5.

$$\frac{1}{|R|}\log\frac{\sum_{t\in T}\exp[-f(R,T)]/|T|}{\sum_{g\in G}\exp[-f(R,G)]/|G|} \tag{5}$$

This formulation captured most of the desirable properties that we sought but it ended up not capturing when the generator missed several modes. A second formulation we considered was the summation of equations 4 and 5. This had all of the desirable properties that the original one had but failed to add anything obvious that the first measure missed. We therefore kept the formulation presented in equation 4 due to it capturing all of the desirable properties and its simpler formulation.

4. Methods

4.1. Datasets

We used the following four datasets and visualize the results as GANs train.

MNIST. The MNIST database [9] of handwritten digits has a training set of 60,000 examples, and a test set of 10,000 examples.

SVHN. The Street View House Numbers (SVHN) dataset [3] is of cropped 32 by 32 digits found through Google Street View. There are 73,257 images in the training set, 26,032 images in the test set, and 531,131 images for additional training.

CIFAR. The two CIFAR datasets [8] consist of colored natural scene images, with 32-by-32 pixels each. CIFAR-10 (C10) consists of images drawn from 10 classes and CIFAR-100 consists of images drawn from 100 classes. For both datasets there are 50,000 training images and 10,000 test images.

ImageNet. The ILSVRC 2012 classification dataset [17] consists of 1000 classes, in total 1.2 million for training, 50,000 for validation, and 100,000 for testing.

4.2. Architectures

For this work we consider two primary GAN architectures for our evaluation, DCGAN [15], and Improved GAN [18]. We modified both architectures to create a weaker generator by removing all of the batch-normalization layers. To create our cherry picked examples for MNIST we removed convolutional layer 4 and modified the third convolutional layer to match the output of convolutional layer 4 from DCGAN. We call this modified architecture small DCGAN. To run our final comparison over models we compared the regular DCGAN, Improved GAN, and Improved Wasserstein GAN [5].

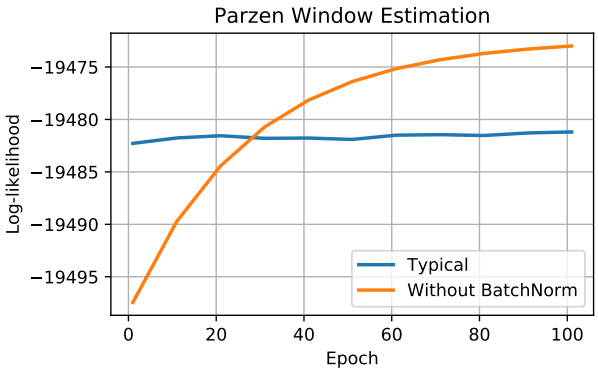


Figure 4: Parzen Window Estimation. Here we demonstrate how the log-likelihood improves over time on the model without Batch Normalization even though the model never achieves better images than the typical model.

For the purposes of creating the embedding we used Wide Residual Networks [21] for both SVHN and CIFAR-10. To embed MNIST we trained a 4 layer CNN architecture, followed by a fully connected layer. We used Xavier initialization [1] for the neural network used to embed MNIST and we used He initialization for the wide residual networks.

5. Experiments

5.1. Comparisons to Other Measures

The first measure we examine is Parzen Windows as shown in 4. The measure shows how the weaker model (the model without Batch Normalization) improves with respect to log-likelihood, yet the samples never achieve anything resembling natural images. This is in contrast to the regular Improved GAN architecture that appears more "natural" over time. Thereby highlighting a results that agrees with [19] that log-likelihood models do not correlate with perceptual image quality.

Additionally Parzen Windows suffers from the curse of dimensionality. Fitting a kernel to the space of natural images results in a very poor estimation as slight changes in the pixel space do not correspond to any natural changes. Leaving much of the space empty resulting in much difficulty estimating the likelihood. Another experiment that we ran was compressing natural images by taking the top principle components that capture 95% of the variation of the data and then running Parzen Windows. In this setting it would be expected that Parzen Windows would perform better but we still observed the same behavior where the weak model outperformed the standard Improved GAN model. The code to run Parzen Windows was obtained from https://github.com/goodfeli/adversarial/blob/master/parzen_ll.py.

With regards to Annealed Importance Sampling in figure 6 we run AIS several times while varying the variance parameter on a fixed GAN architecture, the "GAN10" architecture from the AIS paper. We can observe that the final log-likelihood is heavily dependent on the choice of sigma. This shows how the measure can be manipulated by varying sigma, a hyperparameter, until the desired score is achieved.

It also highlights how the measure is inappropriate for implicit density models as their claims of convergence only hold true for explicit density based generative models of which variational autoencoders are. This is discussed in [20] whereby to estimate the exact posterior one requires simulated data. Therefore the final estimates that they derive can be quite far from the true likelihood except when certain conditions are met of which GANs do not meet.

The Inception Score has a few issues worth mentioning. The first issue occurs when the generated set has a small number of images that resemble ImageNet images. This will generate a high Inception Score, even though the majority of samples are of poor quality. If a generator is able to produce a few samples that activate only one class, then it becomes equivalent to taking the KL divergence between a one hot vector and uniform over 1000 classes. Afterwards one takes the exponential of this quantity so it becomes $e^{\log(N)}$ where $N = 1000$. Implying the maximum inception score is 1000 which is quite far from any of the current datasets even computed on actual images and making the score ultimately difficult to interpret.

Another issue of the Inception Score is that it fails to detect missing modes of the distribution. We created a dataset where we take natural images and replicate them until we match N the number of samples in the training set. By increasing the number of samples we thereby decrease the replication factor of the dataset. The Inception Score reaches its optimal performance once there are 1000 distinct images. Whereas DDM does not reach the optimum with any replication amount.

5.2. Distribution Divergence Measure

In this section we compute the Distribution Divergence Measure, DDM, on three different datasets MNIST, SVHN, and CIFAR-10 and compare the results of the architectures described in section 4.2. We demonstrate that DDM achieves the correct ordinal rankings for each dataset. Due to the architectures having a great dissimilarity in their outputs . We want to start with a baseline task to ensure the model works under supervision before proceeding to more complicated comparisons that may look equivalent to human observers and vary subtlety. We would like to note that removing batchnormalization from the Improved GAN architecture produced the worst results both perceptibly and often several orders of magnitude higher (worse) than all of

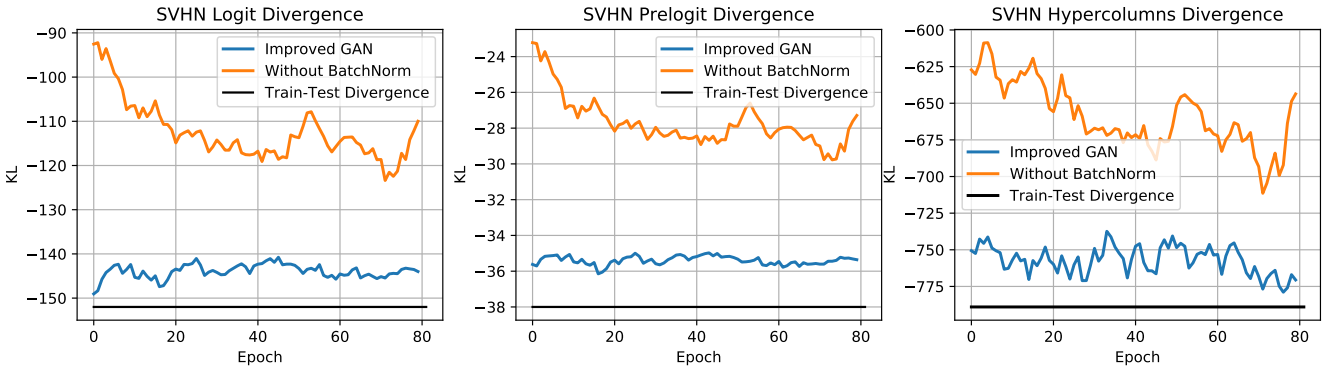


Figure 5: Alternative semantic embeddings. We show that other semantic representations namely logits, prelogits, and hypercolumns track each other with DDM. This highlights that DDM is insensitive to any particular embedding so long as it contains enough semantic information to meaningfully discriminate among the samples.

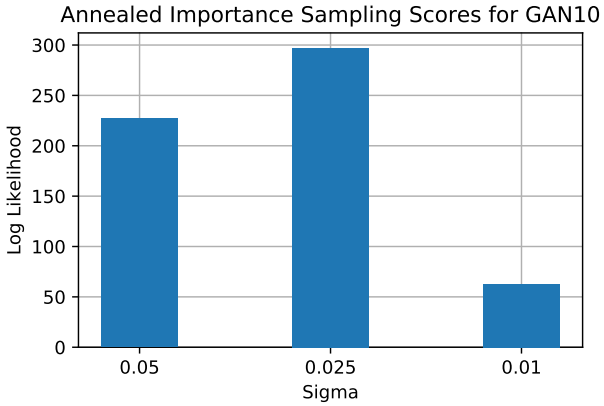


Figure 6: Even for a fixed GAN the Annealed Importance Sampling Score can vary by an order of magnitude depending on the sigma used for the Gaussian prior.

the other architectures, thus they are not plotted.

For our MNIST experiment we tested DCGAN, weakened DCGAN, and small DCGAN. We can observe that the correct ordinal ranking is achieved by the measure, highlighting that the measure detects the missing modes of the distribution. By ranking the small DCGAN better than weakened GAN it highlights that it is not fooled by noise and by ranking DCGAN better than small GAN it further highlights the importance of the full distribution for a better score.

For the SVHN experiment we tested DCGAN, weakened DCGAN and Improved GAN. We left out small DCGAN as we had difficulty determining when small DCGAN was performing better or worse (i.e. producing more visibly natural images) than weakened DCGAN. In our CIFAR-10 experiment we ran all of the same architectures as with SVHN.

5.3. Insensitivity to Embedding

To show that the DDM is not explicitly tied to the logits of a neural network we tested out other representations that contain semantic information encoded from neural networks. In this way we demonstrate that the measure still produces the correct ordering among the compared architectures. We test three different embedding: logits, prelogit, and hypercolumns.

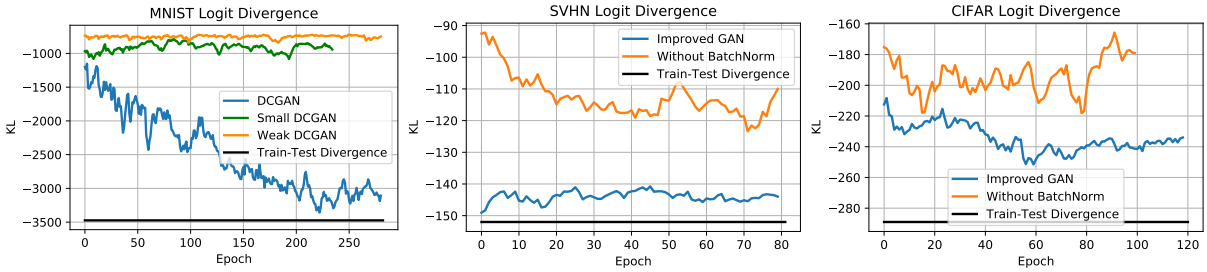
Logits are the values of the final layer of the neural network before applying the softmax function. Prelogits are the inputs to the layer before the logits. Hypercolumns [6] are a sampled representation of each layer in the neural network which contains information from many different receptive fields.

5.4. Insensitivity to Architecture

We also tested whether the measure varies based upon the particular architecture used to create the embedding. We experimented with determining if the measure varied based on architecture used by varying the initialization of each architecture and experimenting with different architectures. Ultimately while the final values from each architecture vary slightly the overall ordering remains unchanged. We would like to note that all of the models have sufficient model complexity to achieve at or near state of the art in the task of classification.

5.5. Missing Modes

Suppose the generative model is not able to capture the full distribution and instead can only generate a few modes repeatedly from the distribution. We test how well DDM and Inception Score can handle this with the following experiment. We take n random images from the training set and duplicate the images in n until we reach the same size as the training set N. Then we run this duplicated subset of the training set through the measure. We take n to be $\{10, 20,$



(a) MNIST results. (b) SVHN results. (c) CIFAR-10 results.

Figure 7: Distribution Divergence Measure plotted over time for the datasets. The baseline for all of the experiments is plotted as the black line. For some datasets we can observe that the model does not learn to generalize much more after the first epoch namely that of Improved GAN on SVHN. The main reason it is likely performing worse under DDM is that it is likely memorizing the training set and therefore becoming less like the test distribution.

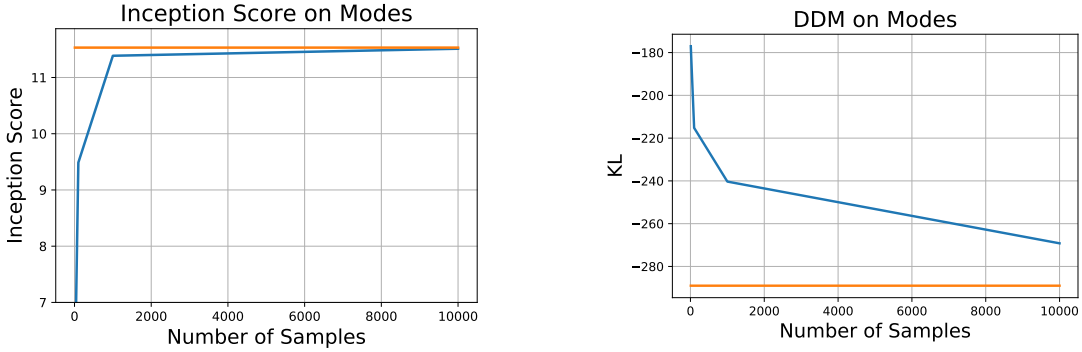


Figure 8: Comparison of Measures on Modes. For this experiment we sampled 10,20,100,1000,10000 images from the training set and replicated the samples up to the size of the original dataset of 50000. The inception score saturates around 1000 images or a a replication amount of 5. For the DDM since we also compare to a test set it does not saturate but instead approaches the relative divergence. For both measures the orange line represents the best performance on each respective measure.

100, 1,000, 10,000} CIFAR-10 training images. To get our final score we repeat this experiment 5 times and take the average of the scores. We show in Fig 8 that the Inception Score approaches its optimum at 1,000 images while DDM does not achieve its optimum as we penalize the model lack of generalization to the test set.

5.6. ImageNet Results

To ensure that the results work for arbitrary sized images we tested DDM on 46,000 images from the ImageNet dataset. We randomly set aside 45,000 images from the validation set that we partition into 40,000 training and 5,000 testing images. We took the remaining one thousand images to serve as our samples. We ran a series of corruptions to the 1,000 image samples, those of additive Gaussian noise, uniform random noise and Gaussian blur. For Gaussian blur with a fixed mean of zero, each additive point to the variance, sigma, roughly corresponds to a loss of 10 bits of information according to DDM. The results for additive noise tended to be on the order of 12 bits.

CIFAR-10 Samples from	Measure Score
DCGAN	−163.14
Improved Wasserstein GAN	−227.66
Improved GAN	−232.74
Training Set	−289.23

Table 1: GAN Scores for various architectures on the CIFAR-10 dataset.

5.7. GAN Generator Comparisons

We wanted to demonstrate that DDM could be used to measure the quality of various GANs, so we chose 3 popular GAN architectures namely DCGAN, Improved GAN, and Improved Wasserstein GAN. We trained each GAN architecture until convergence as deemed their respective implementation on the CIFAR-10 dataset. Then we sample

5000 images uniformly at random from each generator to use as our generated set. See Table 1 for the list of results.

Here we demonstrate that the Improved GAN best approximates the true distribution, while Improved Wasserstein GAN performs slightly worse. The worst performing model is the DCGAN, as expected.

6. Conclusion

We discussed a measure that scores sets of images and works on implicit density models of which GANs are a part. This is the first measure for generative models that detects missing modes, tracks image quality, and works effectively on GANs. We feel that the evidence given is compelling to argue for considering distributions of images rather than per image measures and therefore a divergence measure is most appropriate. We hope the measure presented can be widely used by the community as a competitive benchmark for the task of image generation.

References

[1] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In Y. W. Teh and M. Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. 5

[2] J. Goldberger, S. Gordon, and H. Greenspan. Approximating the kullback leibler divergence between gaussian mixture models. In *International Conference on Computer Vision (ICCV)*, 2003. 3

[3] I. J. Goodfellow, Y. Bulatov, J. Ibarz, S. Arnoud, and V. Shet. Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks. *ArXiv e-prints*, Dec. 2013. 2, 4

[4] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Networks. pages 1–9, 2014. 1

[5] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved Training of Wasserstein GANs. *ArXiv e-prints*, Mar. 2017. 2, 4

[6] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for Object Segmentation and Fine-grained Localization. *ArXiv e-prints*, Nov. 2014. 6

[7] G. Hinton, O. Vinyals, and J. Dean. Distilling the Knowledge in a Neural Network. *ArXiv e-prints*, Mar. 2015. 4

[8] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Tech Report*, 2009. 4

[9] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998. 4

[10] Y. LeCun, F. J. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, CVPR'04, pages 97–104, Washington, DC, USA, 2004. IEEE Computer Society. 2

[11] D. Lopez-Paz and M. Oquab. Revisiting classifier two-sample tests. In *International Conference on Learning Representations (ICLR)*, 2017. 2

[12] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley. Least Squares Generative Adversarial Networks. *ArXiv e-prints*, Nov. 2016. 1

[13] V. Mnih and G. E. Hinton. Learning to detect roads in high-resolution aerial images. In *Proceedings of the 11th European Conference on Computer Vision: Part VI*, ECCV'10, pages 210–223, Berlin, Heidelberg, 2010. Springer-Verlag. 2

[14] E. Parzen. On estimation of a probability density function and mode. *Ann. Math. Statist.*, 33(3):1065–1076, 09 1962. 1, 2

[15] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2016. 2, 4

[16] A. Romero, N. Ballas, S. Ebrahimi Kahou, A. Chassang, C. Gatta, and Y. Bengio. FitNets: Hints for Thin Deep Nets. *ArXiv e-prints*, Dec. 2014. 4

[17] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 2, 4

[18] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Neural Information Processing Systems (NIPS)*, 2016. 1, 2, 4

[19] L. Theis, A. van den Oord, and M. Bethge. A note on the evaluation of generative models. In *International Conference on Learning Representations (ICLR)*, 2015. 2, 5

[20] Y. Wu, Y. Burda, R. Salakhutdinov, and R. Grosse. On the Quantitative Analysis of Decoder-Based Generative Models. *ArXiv e-prints*, Nov. 2016. 1, 2, 5

[21] S. Zagoruyko and N. Komodakis. Wide Residual Networks. *ArXiv e-prints*, May 2016. 5